

vistrička
žunić

uvod u rad i programiranje

commodore 64

Zvonimir Vistrička
Davor Žunić

commodore 64

UVOD U RAD I PROGRAMIRANJE

Narodna tehnika SR Hrvatske

Zagreb, 1985.

PREDGOVOR

Tehnološki napredak čovječanstva doveo je do potpuno novih tehnologija u koje pripadaju i mikroračunala. Po svojoj konstrukciji to je dosta kompliciran uređaj koji sadržava najsuvremenija tehnološka rješenja, no unatoč tome je jednostavan u upotrebi.

Računalo Commodore 64 pripada u klasu mikroračunala, točnije u kućna računala. Vrlo je rašireno u svijetu pa i u nas zauzima drugo mjesto po rasprostranjenosti među korisnicima kućnih računala. Svoje poklonike steklo je niskom cijenom, visokom kvalitetom izrade te vrlo dobrim karakteristikama — memorijom: 64K RAM-a, 20K ROM-a, visokom grafikom, mogućnošću upotrebe 16 boja te trokanalnim zvučnim sintesizerom. To su karakteristike koje zadovoljavaju najšira područja primjene — od igara, obrazovanja pa sve do projektnih proračuna.

Osnovni priručnik koji se dobije uz računalo u nekim je dijelovima manjkav, a dosad dobre literature na našem jeziku za to računalo nije ni bilo. Narodna tehnika kao najmasovnija društvena organizacija koja u nas širi tehničku kulturu odlučila se posvetiti popularizaciji mikroračunala i u izdavačkoj djelatnosti tako da je ova knjižica drugi naslov posvećen računalima. Time će se domaći korisnici Commodorea 64 moći uvesti u rad i programiranje na ovom računalu.

Ova knjižica naravno ne može u potpunosti objasniti i otkriti sve tajne računala C64, ali u nekim elementima znatno nadmašuje slične naslove, posebno u poglavljima o grafici i zvuku. Sadržaj ovog naslova čitatelj mora nadograđivati dodatnom stručnom literaturom.

Zahvaljujemo se kolegama Petru Petroviću i Željku Žitku za stručnu pomoć u pisanju rukopisa.

Zagreb, lipanj 1985.

Autori

UVOD

O MIKRORAČUNALIMA I MJESTO COMMODOREA 64 MEĐU NJIMA

Mikroračunala imaju uz osnovne elektroničke dijelove centralnu mikroprocesorsku jedinicu te dovoljnu količinu korisničke memorije, danas pretežno veličine 64—128K (K = kilobyte = 1024 byta). Tastatura je osnovna ulazna jedinica, a kao osnovna izlazna jedinica služi monitor ili TV-prijemnik. Dakle, mikroračunalo je cijeli kompjuterske sustav u malome, tim više što se priključenjem dodatnih vanjskih ulazno/izlaznih jedinica znatno povećavaju mogućnosti iskorištenja mikroračunala.

Početak razvoja mikroračunala prihvatljivih za široke mase počinje 1975. godine računalom Altair. To nije bilo računalo izgleda na koji smo danas navikli, ali ga je svatko mogao kupiti u dijelovima i sastaviti. Dotad su postojala neka mikroračunala, na primjer tipa PDP namijenjena uskom krugu korisnika — najčešće na fakultetima i institutima.

Godine 1976. u kalifornijskoj Silicijskoj dolini dvojica mladih entuzijasta sastavljaju prvi Apple, računalo čija je osnovica danas najrasprostranjenija u svijetu, te s najvećim brojem vrhunskih programa.

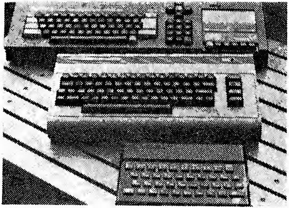
Pravi bum dolazi početkom osamdesetih godina kada tvrtka Sinclair Research počinje lansirati svoja računala ZX 80, ZX 81 i Spectrum te time zauzima prvo mjesto na listi najraspro-

stranjenijih evropskih računala. Pojavljuju se i mnoge tvrtke sa svojim više ili manje uspješnim računalima, a neke velike kompanije propadaju, male napreduju već prema uspješnosti koja se mjeri u relacijama: cijena — programska podrška — kvaliteta. Mnoge velike tvrtke morale su se prilagođavati potražnji svjetskog tržišta, pa u svoj proizvodni program uz velika računala uvrstiti i mikroračunala. Tako i tvrtka Commodore s većih i uredskih računala prelazi na klasu personalnih-osobnih računala (Commodore PET), a potom i na kućno računalo Commodore VIC 20. Računalo VIC 20 je preteča Commodorea 64 koji je slijedio nedugo potom.

Iako s nekim manama (relativno loš BASIC), računalo Commodore 64 odlikuje se sklopovskom kvalitetom, dakle ugrađeni elektronički i mehanički dijelovi su vrlo dobri, a uz to i ogromnom programskom podrškom — ima već na tisuće programa diljem svijeta. Cijenom to računalo pripada među najjeftinije kućne kompjutere, pa uz sve ove karakteristike s pravom nosi epitet najuspješnijeg kućnog računala 1984. godine.

Kako se točno može opisati Commodore 64 u današnjoj konkurenciji? Kao konkurente u klasi kućnih računala treba uzeti Sinclairov Spectrum, Atari 800 XL i Amstrad/Schneider CPC 464.

Usporedimo li Commodore sa Spectrumom može se uvidjeti apsolutna superiornost Commodorea 64 prema Spectrumu, jedino što Spectrum



Commodore 64 nalazi se po svojim karakteristikama između Spectruma i Amstrad/Schneidera CPC 464

ima lakši BASIC i nižu cijenu koja proizlazi iz lošije hardverske opreme. Prema Atariju, pak, Commodore ima veću i jeftiniju programsku podršku te veću raširenost, što u našim uvjetima ima presudno značenje. Uspoređujući Commodore s Amstrad/Schneiderom opet dolazi u obzir sve rečeno za Atari. Amstrad/Schneider je za svoju konfiguraciju izuzetno snažno i jeftino računalo, iako je takva njegova cijena previsoka za naše današnje carinske propise. U našim uvjetima boljeg, jeftinijeg i pouzdanijeg računala od Commodorea 64 nema. Zato ćemo nastojati

ovom knjižicom olakšati upotrebu korisnicima, a ostalima eventualno pomoći da se lakše odluče za kupnju.

Računalo Commodore 64 se može dobiti i u konsignacijskoj prodaji, odnosno od zastupnika KONIM-a, Zagreb, Svačićev trg 14, a osiguran je i službeni servis u Mariboru: Birostroj Maribor. Glavni trg 17b, 62000 Maribor, tel.: 062/ /23-771. Taj zastupnik može nabaviti i dodatnu opremu tvrtke Commodore.

NAMJENA C64 (PODRUČJA PRIMJENE, DOMETI, MOGUĆNOSTI)

Svaka kupnja računala poteže za sobom razne nedoumice, pa vjerujemo da su i sadašnji vlasnici Commodorea 64 bili u dilemi za koje računalo da se odluče. To pitanje ostalo je otvoreno onima koji se tek odlučuju za kupnju računala. Buduće vlasnike zanimaju mogućnosti iskorištenja računala, a u pravilu se odluke uvijek donose nakon usporedbe s drugim računalima i planiranja za što će to računalo služiti.

Pri kupnji je uvijek potrebno postaviti pitanje: Zašto mi računalo treba? i Da li ću se jednog dana više baviti programiranjem?

Na drugo pitanje teže je odgovoriti, ali je prijeko potrebno. Odluka mora najčešće biti dalekosežna jer se može dogoditi da zbog carinskih propisa i lošega kursa dinara budete vezani dugo godina za jedno računalo.

S gornjim pitanjima najčešće se susreću početnici. Obično ih zanima na kojem računalu će se lakše upoznati s osnovima programiranja, odnosno lakše naučiti programirati.

Commodore 64 raspolaže s vrlo ograničenom varijantom programskog jezika BASIC, pa je programiranje početnicima ponešto otežano. Imajući u vidu ovu činjenicu, a ako uz to računalo treba služiti samo za igru, tada se treba odlučiti za računalo tipa Sinclair Spectrum.

Međutim iako vas možda zanimaju samo igre, zacijelo ćete s vremenom poželjeti napraviti neku vlastitu igru. Kada je napravite počeli ste zapravo s programiranjem. Isto tako počet ćete mnoge probleme (matematičke, zbirke podataka itd.) rješavati računalom. Upravo ovdje dolaze do izražaja velike vrijednosti Commodorea 64, u biti vrlo dobrog računala.

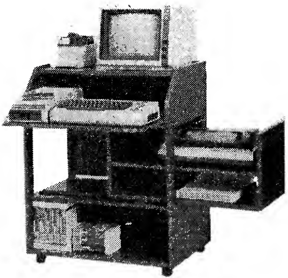
Računalo C64 ima još jednu veliku prednost pred svojim najčešćim konkurentom Sinclairovim Spectrumom — to je visokokvalitetna tastatura standardnog QWERTY-rasporeda. Savlada-

vanjem te tastature stječe se dobra osnovica za poznavanje daktilografije i privikavanja na rad s drugim računalima. Dakle i odavde proizlaze koristi. Za usporedbu, na Spectrumu je također standardni QWERTY-raspored, međutim on zapravo nema praktičnu vrijednost jer se na njemu sve naredbe dobivaju pritiskom na jednu tipku (one keyword-system). To je ponekad praktičnije za početnike, ali izostaje dobro svladavanje tastature.

Zaključak može glasiti: ako vam je potrebno računalo i za naprednije programiranje, a imajući u vidu izbor računala koja se u nas mogu kupiti ili uvesti, odlučite se za Commodore 64.

Računalo koncipirano kao C64 ima velika područja primjene. Ako se krene od igara koji su najbolji mamac računalu treba reći da ih za C64 postoji obilje, i to raznih vrsta. One mogu biti najjednostavnije u stilu da slovo O lovi po ekranu slovo A, zatim razne društvene igre: kartaške, domino, šah, otelo, zatim kojekakve simulacije letova, vožnji, igre ispitivanja refleksa kao i obrazovne igre s matematičkom, povijesnom, jezičnom, zemljopisnom itd. osnovicom. Svatko u tom rasponu igara može pronaći nešto zanimljivo i tako zakoračiti najčešći prvi korak na računalu.

Računalo Commodore 64 nosi naziv kućno računalo. To znači da ono može hardverski (sklopovski) i softverski (programski) obavljati mnoge kućanske zadaće.



Idealan prostor za rad s računalom C64

Hardware je zajedničko ime za sve elektroničke i mehaničke dijelove računala i njegovih vanjskih jedinica, a software je uvriježeni naziv za programe — programsku podršku.

Ta činjenica znači da računalo C64 može kontrolirati posredstvom različitih elektroničkih naprava (međusklopova) npr. paljenje i gašenje

svjetla u kući, dizanje roleta, upravljanje alarmnim uređajem te mnoge druge funkcije.

Programski možemo upravljati tim postupcima ili sačinjavati najraznovrsnije kućanske banke podataka — npr. evidenciju troškova u kućanstvu, obiteljski telefonski imenik, adrese prijatelja, popis gramofonskih ploča, sadržaj kućne biblioteke ili kao posebnu pogodnost za domaćice — knjigu recepata.

Obrtnici mogu pomoću računala voditi svoje knjigovodstvo, stanje zaliha materijala, a ugostitelji mogu pomoću Commodorea izvesti jelovnik, cjenik, evidentirati stanje uskladištenih namirnica te naručivati ili plaćati robu.

Uz pisač (printer) računalo se može pretvoriti u odličan električni pisaći stroj, a uz dobar program za uređivanje teksta (text-procesor ili word-procesor) može poslužiti i u manjoj tiskari. Opremljen uz pisač i disk-jedinicom C64 može vrlo dobro poslužiti kao manji uredski sustav. Na diskete u disk-jedinicama stane vrlo mnogo informacija, dovoljno da na nekoliko disketa sadržava cjelokupno godišnje knjigovodstvo manjeg poduzeća.

Računalo ima naravno i obrazovnu ulogu, ne samo u školskim okvirima već i u tzv. izvanškolskim aktivnostima. Danas se posebno vrednuje mjesto računala u predškolskoj dobi djece, jer programi namijenjeni najmlađoj generaciji potiču logično razmišljanje i razvijaju maštu baš kao nekad lego-kocke.

Uz velike mogućnosti C64 postoje i neka ograničenja, jer nema računala koje bi na poljima svoje upotrebe svugdje davalo jednak učinak.

Relativno skromni BASIC je prvo ograničenje, a to znači da se programer može rijetko kada do kraja »izraziti« bez poznavanja osnova strojnog programiranja, mnogo kompliciranijeg nego što je BASIC. Takav BASIC može biti i znatno ograničenje za ozbiljnije programiranje.

Drugi »najveći« nedostatak je sporo snimanje i učitavanje podataka i programa na Commodoreove dodatne jedinice koje služe kao vanjske memorije — na kazetofon i disk-jedinice. Sporost u radu rezultira ponekad neprikladnošću za krajnje ozbiljne namjene u kojima se zahtijeva vrlo brz pristup podacima i programima.

U nedostatke Commodorea može se ubrojiti i dosta loša kvaliteta otisaka koji se dobivaju na originalnim pisačima.

No, i uza sve ove nedostatke vješt korisnik će moći izvući iz svog računala maksimum. Nedostaci osnovnog BASIC-a mogu se izbjeći upotrebom Simon's BASIC-a koji omogućuje uvođenje stotinjak novih naredbi i time daje vrlo praktičan BASIC.

Sporo učitavanje programa kao i njihovo snimanje izbjegava se na kazetofonima posebnim »turbo«-programima koji omogućuju znatno ubrzanje tih procesa. Što se rada s disk-jedinicama

tiče, problemi se najlakše mogu izbjeći upotrebom novih Commodoreovih disk-jedinica.

Ako želimo izbjeći lošiju kvalitetu otiska pisača (printera), mogu se upotrebljavati također novi pisači iz Commodoreovog proizvodnog programa. Naravno, možemo uzeti i pisače drugih proizvođača, samo pri kupnji treba paziti da li odgovaraju računalu C64.

Tako se nedostaci Commodorea mogu svesti na podnošljivu razinu, pa računalo može poslužiti i za ozbiljne primjene. Treba ipak biti realan i ne tražiti pretjerano mnogo od računala izvan standardnih okvira.

Uz prije spomenute uobičajene upotrebe Commodore 64 sve više služi kao stroj za likovno i glazbeno izražavanje.

Pri likovnom izražavanju pomaže prilagođena programska podrška kao i posebne periferijske jedinice, npr.: elektronička crtaća ploča (digitazing tablet), svjetlosna olovka (light-pen), itd. Prikladnim programima možemo dobiti zanimljivu kompjutersku grafiku i jednostavnu animaciju.

Računalo C64 ima solidne glazbene mogućnosti pa danas mnogi proizvođači nude međusklopove koji omogućuju povezivanje računala s raznim vrstama elektroničkih muzičkih instrumenata. Upotreba likovnih i zvukovnih mogućnosti zapravo je najkreativnija primjena Commodorea 64, pa se stoga mnogi budući vlasnici računala odlučuju za C64 imajući u vidu programsku podršku i sklopove koji omogućuju takvo iskorištenje.

Od najozbiljnijih primjena treba spomenuti da C64 može poslužiti kao vrlo dobar mjerni instrument ili kao naprava za izradu raznih tehničkih projekata.

Za većinu kućnih računala područje primjene za projektne svrhe nedostižan je domet. Iako se u zadnje vrijeme nastoje neke značajke projektiranja i praćenja proizvodnje pomoću računala (CAD/CAM — Computer Aided Design i Computer Aided Manufacturing) prenijeti na sva mikroračunala i kućna računala, ipak zasada u toj klasi prednjači samo C64. To zato što se u SAD i SR Njemačkoj sve više nalaze konkretne primjene Commodorea 64 u tim područjima, odnosno ima vrlo mnogo literature i programa koji služe kao osnovica CAD-a na tom kućnom računalu.

Ovo posljednje moglo bi zainteresirati i domaće korisnike Commodorea, pogotovu kao primjena u »maloj privredi«, gdje bi ponegdje projektiranje mikroračunalom bilo i te kako korisno.

Dakle, Commodoreu 64 može se naći primjena od igračke do ozbiljnog projektnog stroja u nekoj obrtničkoj radionici. Možda je najbolje uz male rezerve prihvatiti krilaticu koja vrijedi za računala: Jedino ograničenje je vaša mašta.

UPOZNAVANJE I RAD S RAČUNALOM

SUSTAV I UPOTREBA C64

Dijelovi i njihova uporaba s kratkim opisom

Pošto ste nabavili računalo, a vjerojatno i originalni kazetofon, razgledajte kutije u kojima su oni zapakirani. Počinjete s otpakiravanjem, ali pripazite da ne biste slučajno dodirivali kontakte na stražnjoj strani računala. Pitate se zašto?

Kontakti vode prema mikroelektroničkim komponentama — integriranim krugovima (čipovima) koji su osjetljivi na statički elektricitet.



Uz osnovnu konfiguraciju (računalo, kazetofon, ispravljač) vrlo često se odmah kupuje i palica za igru (joystick)

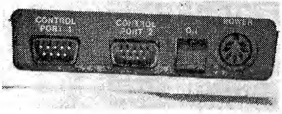
On ih može oštetiti odnosno onesposobiti računalo. Statički elektricitet je gotovo svugdje prisutan pa tako i na vašim rukama.

Iz kutije izvadite računalo, ispravljač, antenski kabel i priručnik za upotrebu. Uz eventualno kupljeni kazetofon također dobivate priručnik. Odložite sve osim samog računala koje razgledajte sa svih strana.

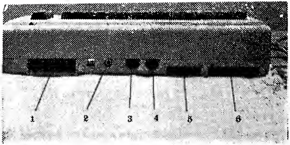
Tastatura je visokokvalitetna. Na gornjoj strani kućišta nalazi se malo signalno crveno svjetlo (LED-dioda) koja pokazuje kada je računalo uključeno.

S desne strane računala nalazimo (sdesna nalijevo): utičnicu gdje se priključuje ispravljač, prekidač za uključenje i isključenje računala (ON OFF) te dva ulaza namijenjena palicama za igru (joystick) označena kao PORT1 i PORT2.

Straga na računalu postoji šest priključnica od kojih svaka ima svoju specifičnu namjenu.



Na kontrolni port 1 može se priključiti i svjetlosno pero (lightpen)



Pregled priključnica sa stražnje strane računala

Promatrajući slijeva nadesno stoje priključnice:

1. Priključnica za module (cartridge) — npr. za one u kojima su upisani programi raznih namjena (igre, upotrebni programi ili prijeko potreban Simon's BASIC).

2. TV-priključnica — mogućnost priključenja računala na običan TV-prijemnik preko antenskog priključka (koaksijalnog kabela od 75 oma). Signal iz ove priključnice prenosi sliku i ton do vašeg televizora.

3. Audioizlaz i videoizlaz — omogućuje da se tonski signal iz računala vodi na Hi-Fi uređaj (visoka vjernost reprodukcije zvuka) te da se kompozitni videosignal (slika) vodi na monitor.

4. Serijski ulaz i izlaz — služi za priključak pisača (printera) ili disk-jedinice.

5. Priključnica za kazetofon — tu ulaze i izlaze podaci koji se spremaju ili učitavaju na kazetofon ili s kazetofona kao vanjske memorijske jedinice. Priključenje standardnoga kazetofona nije moguće bez posebnog međusklopa.

6. Korisnički port (User Port) — programabilni ulaz/izlaz i mogućnost priključivanja raznih međusklopova npr. serijski interface RS 232.

Ponavljamo još jednom — ne dodirujte priključnice rukama kako se računalo ne bi oštetilo.

Priključivanje i puštanje u pogon

Važno upozorenje:

SVAKO ISKLJUČENJE ILI UKLJUČENJE DODATNIH JEDINICA I NAPRAVA DOK JE RAČUNALO POD NAPONOM MOŽE IZAZVATI KVAR! ZATO JE UKLJUČIVANJE I ISKLJUČIVANJE DOPUŠTENO SAMO KADA RAČUNALO NIJE POD NAPONOM! RAČUNALO UKLJUČUJTE I ISKLJUČUJTE SAMO ZA TO PREDVIĐENOM SKLOPKOM!

Računala C64 stižu u našu zemlju od svakuda. Na računala nabavljena u Americi treba posebno paziti jer su tamo druge vrijednosti mrežnog napajanja (110 V / 60 Hz) te drugi standard za televiziju u boji (NTSC). Računala nabavljena u Francuskoj prilagođena su njihovoj kolor tv-normi (SECAM), a ona iz Velike Britanije izazivaju probleme s tonom koji se dobiva s televizora.

Ako smo se uvjerili da računalo zadovoljava naše standarde (napajanje 220 V / 50 Hz) i normu (PAL) možemo razmotriti načine priključenja na vanjske jedinice.

1. Priključivanje na TV-prijemnik

Priključivanje računala na televizor vrlo je jednostavno i češće ga susrećemo. Potrebno je samo antenski kabel koji se dobije uz računalo utaknuti u priključnicu (TV-izlaz) na računalu, a drugi kraj priključiti na televizor tamo gdje se priključuje antena. Normalno, za vrijeme rada s računalom otpada gledanje tv-programa. Spojite i ispravljač s računalom te ga uključite. Sada je na vašem televizoru potrebno ugoditi 36. kanal UHF-područja. Na tom području naše televizijske stanice emitiraju svoje druge programe.

Commodore 64 je »kolor-kompjuter«, pa se stoga predviđa njegova upotreba uz kolor TV-prijemnike. Upotreba je naravno moguća i uz crno-bijeli televizor samo tada izostaje boja na računalu.

Čitatelje ovdje upozoravamo da stari crno--bijeli prijemnici imaju tzv. simetričan antenski ulaz (plosnata žica antene). Tada treba kupiti prilagodni član koji prilagođuje impedanciju modulatora računala (75 oma) na antenski simetrični ulaz televizora (240—300 oma). Prilagodni član izrađuju naši proizvođači pa se nabavlja bez problema.

2. Priključivanje računala C64 na monitor

Preko kolor-monitora dobiva se puno bolja slika nego na TV-prijemniku. Monitor je zapravo kao televizor samo bez ugradenog birača kanala — tunera, kojim se ugađa prijem programa s odašiljača.

Videokabelom utaknutim u priključnicu br. 3 spaja se računalo s monitorom.

Ako je na vašem televizoru posebno izvedena TV-priključnica — možete ga upotrijebiti kao monitor. Potrebno je samo paziti da je isključen videoulaz »prema« računalu koji je inače potreban za rad s kućnim magnetoskopima. U pravilu to se s kućnim magnetoskopima postiže pomoćnim naponom od 12 volti koji postoji na kontaktu 1 videoutičnice televizora.

Commodore 64 nema takav pomoćni napon, pa ga je potrebno ugraditi — to mora učiniti neka stručna osoba. Nestručni zahvati mogu trajno oštetiti računalo. Ako izvodite potrebne zahvate služite se shemom računala i osnovnim priručnikom gdje se govori o tome.

3. Priključak na gradsku mrežu

Postiže se ispravljačem utaknutim u priključnicu s desne strane računala i priključenog na gradsku mrežu (220 V / 50 Hz).

4. Priključak na Hi-Fi liniju

Da bi se postigla veća kvaliteta reprodukcije zvuka može se priključiti C64 na Hi-Fi liniju spojivši računalo preko priključnice AUDIO/VIDEO.

Signali koji izlaze iz priključnice AUDIO/VIDEO mogu se razdvojiti na zvučni signal i signal slike te ih odvoditi na Hi-Fi liniju i monitor.

5. Priključivanje kazetofona

Originalnim Commodoreovim kazetofonom može se upravljati posredstvom računala. Veza računala i kazetofona kao osnovne vanjske memorijske jedinice uspostavlja se priključnicom br. 5 na stražnjoj strani računala. Standardni kazetofoni mogu se priključiti na C64 samo uz pomoć posebnog međusklopa.

6. Priključivanje dodatnih vanjskih jedinica

Priključivanje printera i disk-jedinice postiže se utičnicom br. 4 na stražnjoj strani računala. Tipični model Commodoreova printera je VC 1525, a disk-jedinice (»Floppy Disc«) VC 1541.

Prvi korak na računalu je uključivanje sklopke na desnoj strani kućišta. Trenutak potom ekran će poprimiti ovaj izgled:

++++ COMMODORE 64 BASIC V2 ++++
64K RAM SYSTEM 38911 BASIC BYTES FREE
READY

■

Treptavi kursor (pokazivač) označava da je računalo spremno za rad. Kursor pokazuje na kojem će mjestu biti prikazan prvi znak unesen u računalo.

Boja ekrana je tamnoplava sa svijetloplavim rubom, a slova su također svijetloplava.

38911 BASIC BYTES FREE znači da na raspolaganju imate 38 kilobyta slobodne memorije za vaše programe u BASIC-u.

Slika mora biti točno ugođena kako bi njezina oštrina bila optimalna.

Šest je najčešćih problema koji se ponekad pojave prilikom prvog uključivanja računala. Razmotrit ćemo ih redom:

1. simptom:

Crvena signalna LED-dioda ne svijetli.

Razlozi:

— Računalo nije ni uključeno — uključite ga sklopkom, tj. postavite je u položaj ON.

— Nema kontakta s mrežnim napajanjem — ispitajte utikače na ispravljaču te da li uopće ima struje u zidnoj utičnici. To učinite ispitivačem faze (»probirštiftom«).

— Pregorjeli osigurač u računalu — treba ga izmijeniti. Otvaranjem računala gubite garanciju, ali kako je većina C64 kupljena u inozemstvu i povratak u tamošnji servis je vrlo kompliciran, zamolite nekog stručnjaka da vam izmijeni osigurač koji mora biti iste vrijednosti kao originalni.

2. simptom:

Nema slike na ekranu.

Razlozi:

— Kanal na TV-prijemniku je krivo namješten — ispitajte da li ste odabrali UHF-područje, odnosno da li je 36. kanal točno ugođen.

— Antenski ili videokabel nije spojen ili je u prekidu — spojite kabel pravilno, a ako i dalje nema slike ispitajte spojeve žica na utikačima.

3. simptom:

Loša boja.

Razlog:

— TV-prijemnik nije dobro ugođen — prekontrolirajte ugođenost boje i kanala na prijemniku. Ako imate neevropske varijante računala boju ne možete ni očekivati zbog različitih standarda televizije u boji.

4. simptom:

Jak šum.

Razlog:

— Glasnoća zvuka je na televizoru namještena preglasno — potrebno je stišati ton da nema izobličenja.

5. simptom:

Slika je dobra, ali nema tona.

Razlozi:

— Pretiho namješten ton na televizoru — pojačajte ga.
— Audioizlaz nije spojen na ulaz pojačala — spojite računalo na AUX (pomoćni) ulaz pojačala.
— Posjedujete britansku varijantu računala — potrebna je manja korekcija unutar računala.

6. simptom:

Boje ima, ali se ne pojavljuju slova ni kursor na ekranu kada je priključen dodatni modul.

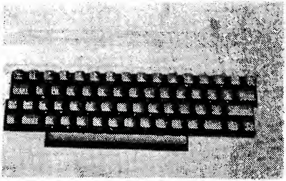
Razlog:

— Modul nije pravilno priključen — isključite računalo i modul pravilno priključite.

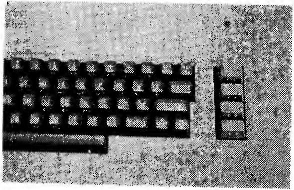
To su tipične »početničke« greške, a nadamo se da niste imali većih problema. Ako je u pitanju kvar, potražite ovlašteni servis ili stručnjaka za kućna računala.

TASTATURA C64

Tastatura C64 u svim glavnim crtama odgovara tastaturi pisaćeg stroja uz dodatak poseb-



Dio tastature računala C64



Dio tastature računala C64

nih. funkcijskih tipki. Njihovu namjenu ćemo objasniti u ovom poglavlju.

TIPKA

RETURN — jedna od najbitnijih na tastaturi, jer se njome daje znak kompjuteru da je naredba ili podatak upisan i kompjuter ga može izvršavati.

SHIFT — ovom tipkom dobivamo velika slova, ako je računalo u modu »velika/mala slova«, ili grafičke znakove, kada je u modu »velika slova/grafika«. Tada, kada istodobno pritisnemo SHIFT i neko slovo, na ekranu dobivamo desni grafički znak s tipke slova. Iz BASIC-a modove uključujemo naredbom POKE 53272,23 za mod »velika slova / mala slova«.

CRSR — te dvije tipke služe za pomicanje kursora po ekranu. U normalnom modu kursor se na pritisak CRSR kreće desno ili dolje (ovisno o tipki), ali ako pritisnemo CRSR i SHIFT zajedno, kursor se kreće lijevo ili gore. Tako možemo dovesti kursor na bilo koje mjesto na ekranu. CRSR-tipke imaju ugrađeni tzv. »auto-repeat«, što znači da će se kursor pomicati sve dok je tipka pritisnuta.

INST/DEL — ovu tipku upotrebljavamo pri ispravljanju (editiranju) programskih linija. Ako pri pisanju učinimo neku grešku, možemo je obrisati pritiskom na tipku INST/DEL. No, ako grešku otkrijemo kasnije, trebamo prvo pozicionirati kursor desno od krivoga znaka (CRSR-tipkom), izbrisati ga s INST/DEL, a zatim da bismo umetnuli novi znak trebamo napraviti mjesto za njega. To činimo tako da pritisnemo SHIFT i INST/DEL zajedno. U liniji će se stvoriti rupa u koju tada možemo upisati ispravak. Tipku INST/DEL i SHIFT trebamo pritisnuti onoliko puta koliko slova želimo upisati.

CLR/HOME — ovom tipkom kursor se postavlja u gornji lijevi kut, a ako pritisnemo SHIFT i CLR/HOME istodobno, očistit ćemo ekran i vratiti kursor u lijevi gornji kut.

RESTORE — u kombinaciji s tipkom RUN/STOP vraća računalo u početno stanje.

CTRL — ovom tipkom biramo boje (CTRL i tipka 1—8 zajedno) ili uključujemo i isključujemo reverzni mod slova i znakova (CTRL i 9 -RVS ON, CTRL i Ø -RVS OFF).

RUN/STOP — ovom tipkom zaustavlja se izvođenje BASIC-programa.

COMMODORE (znak) — ova tipka ima više funkcija. Njome se mogu na ekranu dobiti grafički znakovi koji su lijevo na tipkama. Ako pritisnemo tipke SHIFT i COMMODORE zajedno, prelazimo iz moda »velika slova/grafika« u mod »velika/mala slova«. Funkcija tipke COMMODORE jest da omogući uključivanje još 8 boja na tipkama 1—8.

set 1	set 2	poke	set 1	set 2	poke	set 1	set 2	poke	set 1	set 2	poke	set 1	set 2	poke	set 1	set 2	po
@		0	V	v	22	,		44			64		V	86			1(
A	a	1	W	w	23	–		45		A	65		W	87			.1C
B	b	2	X	x	24	.		46		B	66		X	88			11
C	c	3	Y	y	25	/		47		C	67		Y	89			11
D	d	4	Z	z	26	0		48		D	68		Z	90			11
E	e	5	[		27	1		49		E	69			91			11
F	f	6	£		28	2		50		F	70			92			11
G	g	7	]		29	3		51		G	71			93			11
H	h	8	↑		30	4		52		H	72			94			11
I	i	9	←		31	5		53		I	73			95			11
J	j	10	SPACE		32	6		54		J	74	SPACE		96			11
K	k	11	!		33	7		55		K	75			97			11
L	l	12	"		34	8		56		L	76			98			12
M	m	13	#		35	9		57		M	77			99			12
N	n	14	$		36	:		58		N	78			100			12
O	o	15	%		37	;		59		O	79			101			12
P	p	16	&		38	<		60		P	80			102			12
Q	q	17	'		39	=		61		Q	81			103			12
R	r	18	(		40	>		62		R	82			104			12
S	s	19	)		41	?		63		S	83			105			12
T	t	20	*		42					T	84			106			
U	u	21	+		43					U	85			107			

Znak	Kod
	0
	1
	2
	3
	4
WHT	5
	6
	7
BLOKIRAN C=	8
DEBLOK. C=	9
	10
	11
	12
UKLJ. MALIH SLOVA	13
	14
	15
	16
CRSR	17
RVS ON	18
CLR HOME	19
INST DEL	20
	21
	22
	23
	24
	25
	26
	27
RED	28
CRSR →	29
GRN	30
BLU	31
SPACE	32
!	33
"	34
#	35
$	36
%	37
&	38
'	39
(	40
)	41
*	42
+	43
,	44
–	45
.	46
/	47
0	48
1	49
2	50
3	51
4	52
5	53
6	54
7	55
8	56
9	57
:	58
;	59
<	60
=	61
>	62
?	63
@	64
A	65
B	66
C	67
D	68
E	69
F	70
G	71
H	72
I	73
J	74
K	75

L	76	←	95	●	113		132
M	77		96		114	f 1	133
N	78	♠	97	♥	115	f 3	134
O	79		98		116	f 5	135
P	80		99		117	f 7	136
Q	81		100	☒	118	f 2	137
R	82		101	○	119	f 4	138
S	83		102	♣	120	f 6	139
T	84		103		121	f 8	140
U	85		104	♦	122	[illegible]	141
V	86		105	⊞	123		142
W	87		106		124		143
X	88		107		125	BLK	144
Y	89		108	π	126	CRSR	145
Z	90		109		127	RVS OFF	146
[	91		110		128	CLR HOME	147
£	92		111		129	INST DEL	148
]	93		112		130		149
↑	94				131		150



151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169



170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187



190
191



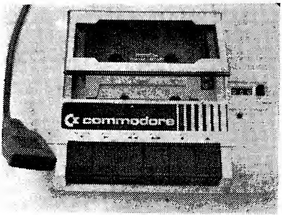
188
189

SHIFT LOCK — pritiskom na ovu tipku funkcija tipke SHIFT se ostavlja stalno na snazi.

Kako se pisanje programa može skratiti tako da se umjesto naredbi upisuju početna slova i grafički znakovi, u nastavku se nalazi lista tih naredbi i grafičkih znakova koji služe pri pisanju programa. Uz to su i kodovi slova i znakova CHR$ i ASCII.

KAZETOFON

Uz C64 najčešće se kao medij za pohranjivanje podataka upotrebljava njegov kazetofon. Kako običan kazetofon ne može poslužiti bez upotrebe posebnog međusklopa, većina se pri kupnji kompjutera odmah odlučuje i za DATASSETE. U usporedbi s običnim kazetofonom on pruža neke bitne prednosti. Kao prvo, potpuno je prilagođen za rad s kompjuterom, te se greške pri spremanju i učitavanju vrlo rijetko javljaju. Uz to, pogon mu kontrolira kompjuter, pa je time olakšano učitavanje i spremanje programa. Kazetofon se također napaja preko višežilnoga ka-



bela kojim je spojen s kompjuterom. Povremeno je potrebno očistiti magnetsku glavu vatom namočenom u čisti alkohol, jer se na njoj skupljaju čestice koje otpadaju s vrpci i prašina. Osnovne naredbe kojima se programi spremaju na kazetofon su SAVE, LOAD, VERIFY, a preciznije su objašnjene u poglavlju o BASIC-u C64.

Na slici se nalaze glavni dijelovi i tipke kojima upravljamo s kazetofonom. Kazetofon se u kompjuter uključuje samo kad je isključen iz napona. U suprotnom mogu se oštetiti i računalo i kazetofon, iako se to ne mora uvijek dogoditi. Za test upisivanja podataka možete upotrijebiti ovaj kratki program:

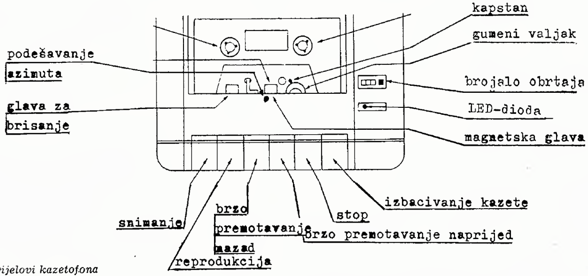
```
1Ø REM — bilo kakav tekst
2Ø REM — bilo kakav tekst
. . . . . . . . . . .
9Ø REM 213456789 Ø
```

Nakon toga upišite direktnu naredbu SAVE »test« i pritisnite RETURN. Na ekranu će pisati poruka PRESS RECORD&PLAY te pritisnite te dvije tipke na kazetofonu. Kad se snimanje završi, premotajte vrpcu na početak i upišite naredbu LOAD »test«, pritisnite RETURN i PLAY na kazetofonu. Program će se učitati i tada ga izlistajte naredbom LIST i ispitajte da li je korektno učitan. Ako imate problema s učitavanjem, najčešći krivac je loše podešen azimut magnetske glave, koju možete sami namjestiti tankim odvijačem kroz rupicu iznad magnetske glave.

DISK-JEDINICA

Za imalo ozbiljnijeg korisnika računala, disk-jedinica je prva stavka u kupnji nakon što je nabavio samo računalo. Disk, zapravo njegov operacioni sistem oslobađa korisnika od zamornog i dosadnog dijela posla oko spremanja programa i podataka. Dok se radi sa kazetofonom, treba se znati gdje je program na vrpci, naći ga, pa tek onda učitavati. Sa diskom se sve te radnje obavljaju pod kontrolom procesora, a uz

to i nekoliko puta brže. Prosječna razlika u brzini učitavanja programa je oko 50 puta, ali to baš i nije tako s COMMODORE-ovom disk-jedinicom VIC 1541, koja je najpristupačnija po cijeni. U SRNj stoji oko 700 DM. Na C64 mogu se priključiti do 4 takve jedinice odjedanput. Kapacitet diskete je 170 K, tj. do 144 fajla (niza). Disk se s računalom povezuje šestožilnim DIN-kabelom, koji se dobiva pri kupnji. Izvor napajanja je smješten u istom kućištu s diskom, pa je dovoljno da se priključi na mrežu s kabelom. Uz disk se dobiva i demonstraciona disketa (5,25 inča) na kojoj se nalazi desetak programa koji su vrlo korisni u radu s disk-jedinicom. Sam disk je »inteligentan«, jer se u njemu nalazi još jedan procesor 6502, 16 K ROM i 2 K RAM, što omogućava programiranje procesora u disku. Jedna je od prednosti takve konfiguracije u tome



Dijelovi kazetofona

što ne zauzima memorijski prostor samoga kompjutera, jer mu je i DOS-disk operacioni sistem ugrađen u ROM-u. Na nekim drugim sistemima, DOS se mora učitavati s diska u RAM, pa se time gubi dio korisničke memorije.

U zadnje vrijeme pojavile su se nove, brže disk-jedinice (naravno, i skuplje) za C64, a i neki alternativni proizvodi poput QUICKDISK-a ili WAFADRIVE-a, koji su jeftiniji od 1541, no imaju brži pristup podacima, pa je i to korisno uložen novac.

PISAČ (PRINTER)

Za COMMODORE 64 ima prilično širok izbor pisača i plotera (crtača) koje proizvodi sam COMMODORE i neki drugi proizvođači. To su većinom matrični printeri koji su po cijeni dosta pristupačni (600—1000 DM), a daju relativno korektan otisak na papiru. Primjer su programi u ovoj knjizi koji su izvučeni na pisaču SEIKOSHA GP100 koji ima matricu karaktera 7 × 5 točkica. Brzina tiskanja je oko 50 karaktera u sekundi, a tiska se na formatu papira od 10 inča (A4 format), koji mora biti perforiran. Inače, SEIKOSHA GP100 je identična Commodoreu 1525. Bolji model je VIC 1526, ima vrlo dobar set karaktera, a štampa dvosmjerno, pa mu je i brzina veća od 1525. Može upotrebljavati neperforirani papir ili obični papir A4.. Oba printera mogu skidati i grafiku uz malu pomoć naredbe COPY, koja se nalazi u SIMON'S BASIC-u. U zadnje su se vrijeme pojavili novi Commodoreovi modeli 802 i 803, koji se mogu nabaviti i u nas u zastupništvu.

MODULI

COMMODORE 64 na svojoj stražnjoj strani ima i port za priključenje modula (cartridge). To su zapravo PROM-memorije u koje su upisani programi i pri stavljanju modula u port odmah nam stoje na raspolaganju. Dodatna prednost modula je u tome da ne oduzimaju memoriju, pa ostaje više prostora za podatke. Kako im je cijena relativno visoka, na njih se spremaju programi za profesionalne namjene, a igre se rjeđe sreću. Dobar primjer upotrebe modula je za programe SIMON'S BASIC, TURBO TAPE, za obradu teksta itd.

OSTALA OPREMA

Od ostale opreme koja se može na C64 priključiti treba kao prvo spomenuti palice za igru (JOYSTICK) za koje postoje već predviđena 2

ulaza — PORT 1 i PORT 2. Mogu se upotrijebiti sve tzv. ATARI-palice sa D-konektorom na kraju. Sličnu namjenu ima i PEDDLE, koji ima 256 raznih položaja, za razliku od palica koje imaju 9 položaja.

Također se direktno može priključiti i svjetlosno pero (LIGHT PEN) koje omogućuje crtanje po ekranu, a uz neke programe i mnoge druge radnje.

Ako želite pomoću C64 kontrolirati sintesizer, to možete učiniti pomoću MIDI-interfacea, koji je standardiziran i većina elektroničkih muzičkih instrumenata je prilagođena za njega.

Uz pomoć MODEM-a možete razmjenjivati programe telefonom, ili se uključiti u banku podataka, ili ćete čak moći gledati radio-teletext koji će se emitirati uz radio-program (OMLADINSKOG RADIJA iz Zagreba) i primati razne informacije na ekranu.

Broj primjena kompjutera je ograničen samo maštom i kada bismo ovdje samo nabrojili sve moguće dodatke koji postoje za C64, ispunili bismo pola knjige (tko zna, možda i više). Zato, neka ovo posluži kao osnovna informacija na temelju koje ćete lakše odrediti područje svog interesa u radu s kompjuterom.

GRAĐA I FUNKCIONIRANJE C-64

OSNOVNI POJMOVI — bit, byte, ROM, RAM, EPROM

Budući korisnici računala kao i novi vlasnici susreću se nerijetko po prvi puta s nekim novim pojmovima. Već prvim pogledom na karakteristike Commodorea 64 mogu se uočiti pojmovi: digitalno računalo, bit, byte, ROM, RAM itd.

Rad digitalnih računala u koja pripada i C64 zasniva se na promjenama dva stanja električkih veličina — ima li signala ili nema. To su električki signali prikazani kao logičke jedinice (1) — što znači da postoji neki nivo signala, te kao logičke nule (0) što znači da je impuls signala izostao.

Analogna računala su naprave za pretvorbu jedne fizikalne veličine u drugu. Stvarni proces zamjenjuje se analognim — sličnim. Tako se na primjer protok vode može prikazivati kao protok električne struje. Dakle u analognim računalima neki se fizikalni proces prikazuje drugim koji je lakši za razumijevanje. Nedostatak analognih računala je u jednoznačnosti primjene, a prednost u cijeni koja je u pravilu još niža od digitalnih računala. Analogno računalo može biti i najobičniji termometar, a digitalna su računala kalkulatori ili dječje računaljke s kuglicama.

Što je zapravo analogni signal, a što digitalni?

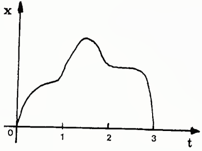
S tim pitanjem u novije se vrijeme susreće sve više ljudi, jer se cjelokupna moderna tehnologija zasniva na digitalnim sklopovima i napravama.

Analogni signal je vrlo lako predočiti ako si zamislimo da iz naše sobe želimo odaslati poruku-informaciju svjetlom do susjedne zgrade. Za to nam može poslužiti danas već uobičajeni svjetlosni regulator na prekidaču — promjenljivi otpornik kojim možemo podešavati željeni intenzitet svjetla. Dogovorimo se s našim susjedom da ćemo sekundu držati srednje slabo svjetlo, sekundu vrlo jako i sekundu malo slabije svjetlo. a da ta »poruka« znači npr. »Dođi k meni«.

Promotrimo ovo gornje sa dva aspekta. U prvom redu, vrlo je teško odrediti što smo zapravo mislili pod jačim i slabijim svjetlom. Svaki čovjek ima različit osjećaj za svjetlo i primatelj poruke bi mogao naš signal shvatiti ne kao dogovoreni poziv već kao nešto drugo. Dakle, mogućnost greške je dosta velika.

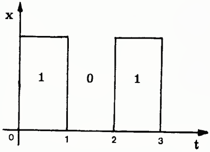
S druge strane, razine osvjetljenja su se mijenjale kontinuirano — stalno, bez naglih prekida.

Pri slanju poruka u svakodnevnom životu, a tako i u komunikaciji s računalom, želimo svesti greške na najmanju moguću razinu te imati oblik signala što praktičnijeg oblika. Takvi sig-



a)

Oblik analognog signala



b)

Oblik digitalnog signala

nali su upravo digitalni signali. Ako gledamo razliku prema prethodnom primjeru, možemo se sa susjedom dogovoriti da će sekundu **biti** svjetla, sekundu **neće biti** i sekundu opet **biti** svjetla. Takvu razmjenu podataka možemo izvesti najobičnijom sklopkom za svjetlo, a upravo mnoge sklopke (elektroničke) postoje unutar našeg računala. Tako se smanjuje mogućnost greške jer su impulsi svjetla jednoznačno određeni. Upravo s takvim nizovima digitalnih signala rade naša računala.

Informacija je subjektivan doživljaj podataka, a podatak kada ga promatramo u okviru računala jest informacija ili program koji se pohranjuju u računalu ili izvršavaju pomoću računala. Najmanja jedinica podatka naziva se BIT, a bit može biti logička jedinica ili logička nula (1 ili Ø). Brzina prijenosa podataka izražava se u BAUD-ima (bodima). Broj bauda odgovara prenesenom broju bita u sekundi. Pojam brzine prijenosa podataka treba nam najčešće kada se govori o vanjskim jedinicama — kazetofonima i disk-jedinicama.

Bitovi su premala količina podataka pa ih treba udruživati. Tako osam bitova čini BYTE (bajt). S osam bitova moguće je postići 256 različitih kombinacija logičkih nula i jedinica ($256 = 2^8$ — dva na osmu potenciju). S tih 256 različitih kombinacija mogu se prikazati brojevi od 0 do 256 ili znamenke, slova, posebni znakovi te instrukcije. Programi sadržavaju veliki broj bytova, pa se za osnovnu veličinu memorije uzima 1 kilobyte (K). On iznosi točno 1024 byta (2^{10}).

Memorije služe za pohranjivanje podataka i programa. U osnovi ih u svakom računalu dijelimo na memorije ROM i RAM.

Memorija ROM (Read Only Memory) je trajno zapisana memorija. Nju si možemo zamisliti kao dio čovječjeg mozga u kojemu su »zapisane« osnovne fiziološke funkcije, upravljanje radom srca i pluća, fiziologijom refleksnih pokreta te obrada informacija primljenih od naših osjetila.

Dakle, memorija ROM ima u sebi trajno zapisan program koji prevodi računalu naše zahtjeve, ali isto tako i zahtjeve računala prevodi u nama prihvatljiv oblik. U memoriji ROM sadržan je BASIC-interpreter koji prevodi riječi BASIC-a u riječi potpuno razumljive računalu (nizove nula i jedinica). U memoriju ROM ne možemo nikako upisati naše programe, niti joj bilo kako promijeniti sadržaj osim potpunog fizičkog uništenja. Njezin sadržaj ne briše se prestankom napajanja, odnosno po isključenju računala. Program koji se nalazi u memoriji ROM zapisan je u toku proizvodnje. U prvim koracima proizvodnje rabi se PROM (Programmable ROM).

Memoriju PROM je moguće izbrisati električki i ponovno u nju upisati popravljeni program. Kada se pokaže potpuna ispravnost programiranog PROM-a pristupa se tek tada serijskoj pro-

izvodnji memorija ROM koje će biti ugrađene u računala.

Posebna vrsta su memorije EPROM (Erasable PROM) koje se mogu lako programirati ali i brisati pomoću ultraljubičastog svjetla. Takve memorije EPROM su posebno pogodne za punjenje raznim programima, pa ih za računalo Commodore 64 možemo naći u obliku modula (cartridge) koji se priključuju na računalo.

Druga vrsta memorije su memorije RAM (Random Access Memory), memorije sa slučajnim pristupom. Memorija RAM je korisnička memorija u koju upisujemo naše podatke i programe. Na računalu Commodore 64 po uključenju ostaje slobodno otprilike 38 K memorije RAM.

Memoriju RAM možemo zamisliti kao dio našeg mozga, naše memorije, koji memorira informaciju kako da na primjer riješimo određeni matematički zadatak, igramo šah ili što drugo.

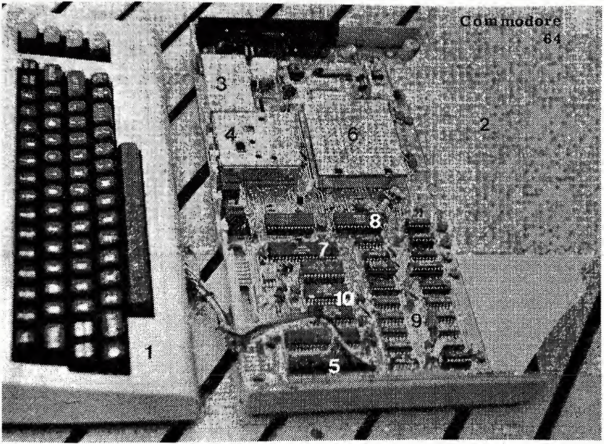
Sadržaj memorije RAM nepovratno se briše po prestanku napajanja. To znači, ako upišemo program te isključimo i ponovno uključimo računalo, u memoriji RAM neće ostati niti jedan dio našeg programa. Upravo zato pribjegava se upotrebi »vanjskih« memorija — memorija EPROM, kazetofona i disk-jedinica. To zato da se programi ne bi morali svaki put ponovno utipkavati tastaturom.

Memorije RAM i ROM izvode se danas najčešće kao poluvodičke memorije, ugrađene u tzv. kućišta dual-in line. Mnogi korisnici računala imali su prilike vidjeti neki memorijski čip bilo na slikama ili pri otvaranju njihovih računala. Ti integrirani krugovi su kao elektroničke komponente dosta veliki, međutim prava je površina čipova znatno manja — u pravilu uvijek manja od 1 cm². Sve je ostalo kućište, odnosno ambalaža tog čipa.

Čip je komadić poluvodičkog materijala, najčešće silicija, na kojem je izveden cijeli elektronički sklop — na primjer memorija, mikroprocesor, zvučni čip itd. Čip se dobiva tehnološkom obradom iz wafera.

Wafer je okrugli disk poluvodiča veličine ljudskog dlana na kojem se odjednom napravi i do 200 čipova. Po završenoj tehnološkoj obradi čipovi se međusobno odvoje rezanjem, te se svaki ugrađuje u svoje kućište.

Svi čipovi su osjetljivi na statički elektricitet koji ih može trajno uništiti. Statički elektricitet najčešće možemo uočiti pri skidanju odjeće od sintetskih materijala ili pri milovanju pasa i mačaka. Statičkog elektriciteta ima svugdje, pa stoga treba paziti da pri eventualnom otvaranju računala ne dodirujemo čipove, ili na zatvorenom računalu konektore-spojnice za priključenje dodatnih naprava. Isto tako ne smiju se uključivati uređaji dok je računalo pod naponom.



gled u otvoreno računalo Commodore 64

IZVEDBA C64

Commodore 64 je u vrijeme kada se pojavio donio u sebi mnoge novosti: grafiku niske i visoke razlučivosti s upotrebom do 16 boja, sprite (sprajt) grafiku pogodnu za dinamičnije prikaze na ekranu i animaciju, te vrlo dobar trokanalni zvučni sintesizer. Sve ove značajke upravljane su iz BASIC-a.

Moderne tendencije razvoja kućnih računala nastoje sve dijelove svesti na što manji broj elektroničkih komponenti, ali to ima za posljedicu da računala (npr. Sinclairova) rade sporije no što bi zaista trebala raditi.

Računalo C64 je koncipirano tako da ima odvojene sklopove za grafiku, zvuk i druge korisne namjene. Tom činjenicom unatoč sporom mikroprocesoru 6510 računalo vrlo brzo obrađuje podatke. Mikroprocesor 6510 je zapravo poboljšani procesor 6502 koji nalazimo u Appleu ili u domaćem Orlu.

Tiskana pločica Commodorea je izuzetno kvalitetno izvedena i riješena. Tvrtka Commodore je i inače poznata po hardwareu visoke kvalitete.

Pogledajmo na slici kako izgleda rastavljeni C64. Kada tastaturu (1) odvojimo od kućišta prvo možete vidjeti zaštitnu masku (2). Malo kućište za module (3) prihvaća programe upisane u dodatnu memoriju ROM. Limena »kutijica« (4) je modulator koji oblikuje sliku za TV-prijemnik. S lijeve strane računala nalazi se konektor za tastaturu (5). Čip VIC (Video Interface Controller- 6) nosi oznaku 6567, a upravlja prikazima na ekranu te je posebno značajan u sprite-grafici. Mozak računala je mikroprocesor 6510 (7). On je odgovoran za cjelokupan rad računala i vanjskih jedinica. Čip SID (Sound Interface Device 8) kontrolira i upravlja zvukom, a nosi oznaku 6581. Čipovi korisničke memorije RAM (9), pohranjuju podatke i naše programe. Prazne se kada isključimo računalo. Čipovi sistemskog programa (10) napunjeni su programom koji omogućuje povezivanje rada svih dijelova računala C64.

Ostali elementi mogu se nazvati popratnom elektronikom, potrebnom za normalno funkcioniranje i medusobnu vezu svih dijelova računala, kao i prilagodbu na vanjske jedinice raznih namjena.

PODJELA MEMORIJE

Već je u nazivu računala istaknuto da kompjuter ima maksimalnu memoriju od 64K kojoj mikroprocesor 6510 može neposredno pristupiti. Kao i sva ostala računala, C64 ima također memoriju ROM s programima koji moraju biti stalno na »dohvatu«.

Ti programi sadržavaju operacijski sustav te programski jezik BASIC. ROM mora biti sadr-

žan unutar adresnog prostora mikroprocesora (64K). BASIC i operacijski sustav zauzimaju 16K memorije. Postoji još 4K »nevidljive« memorije ROM. koju korisnik »ne vidi«, a ona zauzima mjesto u korisničkoj memoriji RAM na adresama 4096 do 8192 (decimalno).

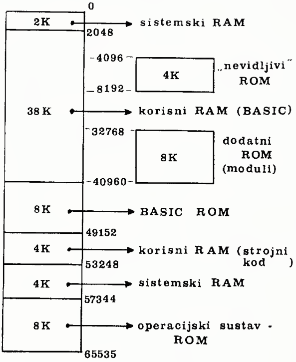
Što su memorijske adrese? Zamislimo stambenu četvrt s mnogo kuća, a time i mnogo stanara. Svaka kuća u četvrti ima svoju adresu (neka nam je to memorijska lokacija), a na toj adresi žive stanari (za nas podaci). Skup stanara na različitim adresama tvori zajednicu koju možemo zamisliti programom jer je i program skup nekakvih smislenih podataka odnosno instrukcija.

Vratimo se na memorije računala. U praksi se može »uključiti« i »isključiti« ROM izvan memorijske mape da bi se dobilo više mjesta za memoriju RAM. Taj postupak je vrlo kompliciran pa se u ovoj knjižici neće objašnjavati.

Promatrajući dalje memorijski raspored može se ustanoviti da korisničke memorije ostaje sve manje.

Dosada se već »izgubilo« 16K RAM-a zbog memorije ROM, a niti cijela memorija RAM nije predviđena za upisivanje programa. Otpada i 6K RAM-a koji je potreban za BASIC i operacijski sustav — KERNAL, točnije za spremanje sistemskih informacija.

Ostaje nam samo 42K RAM-a za upis programa. Međutim samo 38K nam ostaje slobod-



Podjela memorije u računalu Commodore 64

nih za BASIC, i to zato što se u ovih preostalih 4K ne mogu smjestiti programi u BASIC-u već samo npr. u strojnom kodu (machine code).

Standardna memorijska mapa sadržava dakle 16K ROM-a, 6K sistemskog RAM-a, 38K RAM-a za BASIC i 4K RAM-a u kojem se može primijeniti npr. strojni kôd.

Ove činjenice ne treba shvatiti kao nedostatak računala C64 jer ta količina memorije daje dovoljno prostora za raznovrsne upotrebe računala C64 — za kućnu primjenu, igre, grafiku, glazbu itd.

Za neke primjene mogu se dodati posebni paketići ROM (cartridge). Danas se nastoji manje-više svi upotrebni programi kao što su npr. oni za utvrđivanje teksta, jezični prevodioci, assembleri, Simon's BASIC pa i igre smjestiti u memorijske čipove ROM-a. Time se izbjegava klasična programska podrška u obliku kazetofonskih vrpci ili disketa, a ujedno i ubrzava rad računala. Ti paketići zauzimaju 8K na vrhu memorijskog područja BASIC (adrese 32768 — 40959 decimalno) pa imaju obilježja standardnog RAM-a.

Objektivno gledajući, Commodore 64 daje vrlo mnogo memorijskog prostora za malo novaca. I to je jedan od razloga zašto je C64 dobio mnogo oduševljenih pristalica.

UVOD U PROGRAMIRANJE

BASIC — STRUKTURA I NAMJENA PROGRAMSKOG JEZIKA

Po pravilnom priključivanju računala Commodore 64 na mrežu i prijeko potrebne vanjske jedinice, korisnik računala — možda budući ozbiljni programer — susreće se s pojmom programskog jezika BASIC.

Što je BASIC? To je pitanje koje se nameće mlađim korisnicima i ljudima koji se po prvi put susreću s tim pojmom.

Još u vrijeme dok su računala tek zauzimala svoje mjesto u ljudskom življenju javila se potreba za prikladnim programskim jezikom opće namjene, koji će biti razumljiv i lak za uporabu širokom krugu ljudi — od djece do znanstvenika.

Tako sredinom šezdesetih godina na američkom koledžu Darmouth, John Kemeny i Thomas Kurtz stvaraju osnovicu programskog jezika opće namjene — BASIC-a (**B**eginners **A**ll purpose **S**ymbolic **I**nstruction **C**ode).

BASIC pripada među najlakše više programske jezike. Viši programski jezici su pogodniji za upotrebu zato što za naredbe upotrebljavaju cijele riječi ili skraćenice iz svakodnevnog govora odnosno jezika. U području primjene računala dominantan je engleski jezik, tako da su sve riječi instrukcija izvedene iz njega.

Nerijetko se neiskusni korisnici zapitaju zašto se ne mogu upotrijebiti programi pisani u

BASIC-u s nekog drugog računala, pa čak i istog proizvođača. ali različitog modela.

Odgovor na to pitanje mogao bi biti višestruk, ali se može svesti na dvije osnovne činjenice. Prva leži u različitoj strukturi same elektronike u računalu, tj. moglo bi se reći da dio za »objašnjavanje« naših želja računalu nije u svim računalima jednak. On se nalazi u memoriji ROM računala, samo u svakoj vrsti računala je memorija ROM drugačije izvedbe, a i »objašnjenja« su na različitim adresama. To znači na primjer, da u računalu C64 naredba PRINT, tj. njezino objašnjenje leži na toj i toj adresi, a npr. u Sinclairovom Spectrumu na nekoj drugoj adresi. I pošta, iako je vaše ime pravilno napisano, neće biti uručena vama ako je na pošiljci naznačena kriva adresa.

Drugi razlog nemogućnosti primjene programa s drugih tipova računala leži u varijantama jezika BASIC. Danas svaki proizvođač računala upotrebljava neku svoju varijantu BASIC-a, iako postoji »neslužbeni standard« — Microsoftov BASIC. Naravno da računalo neće razumjeti u potpunosti varijantu drugog računala. Isto tako i broj naredbi varira ovisno o varijanti. Prosječni BASIC ima 40—60 instrukcija, a ima računala na kojima se taj broj penje i do iznad stotine naredbi i funkcija. Iz razvoja računala vidi se da su se i programski jezici razvijali usporedno s njima.

Ove činjenice mogu se objasniti i jednostavnije. Svi jezici kojima govori čovječanstvo potekli su iz svojih skromnih korijena, jednostavne i šture osnovice. Razvojem ljudskog društva jezici su se obogaćivali i mijenjali ili dopunjavali.

Na primjeru grčkog jezika može se to lijepo vidjeti. Stari Grci nisu imali riječi za npr. zrakoplov, podmornicu itd., i to zato što takve naprave nisu ni postojale u ono doba. Danas grčki jezik ima nazive za te pojmove, a stoljećima se i mijenjao. Tako se i BASIC od svog postanka mijenjao i prilagođavao tehnologiji računala.

Varijante ili dijalekti BASIC-a mogu se objasniti poznavanjem dijalekata unutar našeg hrvatsko-srpskog jezika. Nerijetko neka riječ ima iako oblikom ista, različito značenje u nekom drugom dijalektu.

I u BASIC-u se mogu pronaći takve razlike. Isto tako neki su dijalekti bogatiji riječima, baš kao i neke varijante BASIC-a.

Naravno da rad na Commodoreu 64 ne zahtijeva poznavanje svih dijalekata BASIC-a, ali poznavanjem više dijalekata, baš kao i poznavanjem dijalekata svakodnevnoga govora, lakše ćemo se moći sporazumijevati, a za računalo to znači — lakše preraditi programe BASIC s nekog drugog za naše računalo C64.

Savladavanjem BASIC-a postajete kompjuterski pismeni. Znati kako se slažu i poredaju

instrukcije jest umijeće koje će u bliskoj budućnosti dobiti puno obilježje prijeko potrebne pismenosti.

Spomenute instrukcije dijelimo na sistemske i programske. Sistemske instrukcije omogućuju rukovanje programima unutar sustava računala. Značajne su posebno za rukovanje s turboprogramima i radu s disk-jedinicama. Sistemske instrukcije daju ujedno odgovore na pitanja kada će se program početi izvršavati?, kada će se prekinuti?, kamo će se program spremiti?, da li da se izbriše?, kako program izgleda? itd.

Za razliku od sistemskih, programske instrukcije daju potrebne informacije računalu. Na bazi tih informacija i podataka računalo će riješiti zadani zadatak.

Programske instrukcije se izvršavaju tek kada se i program nalazi u izvršavanju. Programi su sastavljeni isključivo od programskih instrukcija. složenih u nekom logičkom poretku prema zahtjevima koje postavljamo računalu.

U nekim okolnostima sistemskim instrukcijama dajemo obilježje programskih, samo tada u BASIC-u moraju ispred sebe nositi adresu (redni broj).

OSNOVI BASIC-a

Rad na svakom računalu, pa tako i na Commodoreu 64 zahtijeva upoznavanje s osnovima programskog jezika, ovdje BASIC-a.

Već je spomenuto da su programi logičan niz instrukcija koje omogućuju smisleno rješavanje raznih zadataka, igranje, vođenje evidencije itd. pomoću računala. Instrukcije se mogu podijeliti na: operacije, naredbe i funkcije.

U operacije se ubrajaju matematičke, poredbene i logičke operacije. U naredbe se ubrajaju komande kojima ćemo upravljati računalom, a funkcije dijelimo na numeričke, string i posebne funkcije. U programima se redovito isprepliću sve tri vrste tako podijeljenih instrukcija. Sjetimo se da drugi način podjele dijeli instrukcije na programske i sistemske.

U BASIC-u je potrebno kao i u svim ostalim programskim jezicima naći najkraći i najbolji oblik programa. To u praksi znači da se pravilnom konstrukcijom programa štedi na ionako skučenim memorijama kućnih računala, a kraći program u pravilu znači i brže postizanje cilja — rezultata programa.

Kako se pišu programi?

Programske instrukcije pišu se jedna ispod druge zato da među njima ne bi izbila zbrka. Ispred svake instrukcije nalazi se njezina adresa, odnosno redni broj programske linije U programima je poželjno instrukcije odvojiti većim razmakom, tj. pisati redne brojeve u razmaku od po 5 ili 10 brojeva.

To se radi zato što je ponekad u program potrebno umetnuti poneku instrukciju radi po-

pravljanja ili dopune programa. Kad bi brojevi programskih linija bili dani bez razmaka izazvali bi komplikacije jer se ne bi moglo ništa umetati, a to znači da bi cijelom programu trebalo promijeniti brojeve programskih linija — renumerirati program.

Primjeri:

nepraktično pisanje programskih linija:

```
2 A= 2.4
3 C= 3.1
4 D= A+C
5 PRINT D
```

praktično pisanje programskih linija:

```
 5 A= 2.4
15 C= 3.1
25 D= A+C
35 PRINT D
```

Vidi se da u primjeru praktičnog pisanja postoji razmak od 10 brojeva kako bi se u te razmake eventualno mogle dodati neke dodatne instrukcije, a time proširiti ili popraviti program.

U računalu C64 može se ponekad izvesti »skraćivanje«, odnosno dvije programske linije pisati u jednoj.

Normalni oblik pisanja izgleda npr.:

```
   .
   .
15 POKE 53281,Ø
2Ø POKE 5328Ø,Ø
   .
   .
```

a ako se izvede »skraćivanje«:

```
   .
   .
15 POKE 53281,Ø : POKE 5328Ø,Ø
   .
   .
```

Takvo pisanje ima odredene prednosti, tj. brže će teći dijelovi programa.

Pri upisivanju programa potreban je oprez. Može se dogoditi da se upišu dvije programske linije istog rednog broja. Računalo će tada na dotičnoj adresi (rednom broju) prihvatiti samo posljednju upisanu instrukciju. To se ponekad može izvesti namjerno kao brži način ispravljanja cijele programske linije.

Na primjer, ako se prvo u liniju 55 utipka instrukcija PRINT »C64« dobit ćemo programsku liniju oblika:

55 PRINT »C64«

Ako nakon toga u liniju 55 upisujemo neki novi, različit sadržaj npr. PRINT »COMMODORE 64« — nastaje programska linija:

55 PRINT »COMMODORE 64«

koju će računalo pamtiti kao jedinu programsku liniju rednog broja (adrese) 55.

Da se poveća preglednost programa ili programskih listi, mogu se ubaciti komentari tamo gdje je to potrebno. Svaki program je obično podijeljen u više dijelova: uvod, dio za upis podataka, dio s programskim petljama za daljnje usmjeravanje toka podataka te dio za prikaz

rezultata. Na početku svakog dijela uputno je dati komentar s naznakom što se zbiva u tom dijelu programa.

Komentari se upisuju uz instrukciju REM. ali sadržaj iza te instrukcije računalo ne prihvaća za obradu. Dakle, taj sadržaj ostaje samo radi veće preglednosti napisanih programa.

Primjer:

```
 3 REM PROGRAM ZA ZBRAJANJE DVA
   BROJA (A+B)
 5 REM UNESI VRIJEDNOST A
 7 INPUT A
 9 REM UNESI VRIJEDNOST B
11 INPUT B
13 REM DOBIVANJE REZULTATA
15 C= A+B
17 REM ISPIS REZULTATA NA EKRANU
19 PRINT C
```

Pri pisanju programa srest ćete se s još mnogo pravila, a sva je nemoguće iznijeti na jednom mjestu. Dobro savladavanje svih instrukcija na vašem računalu C64 mora biti osnovno polazište za razumijevanje pisanja vlastitih programa.

Brojevi i njihovo pisanje

Što se znamenaka tiče, jedina je razlika prema klasičnom načinu izgled nule (prekrižena nula — Ø). Taj oblik obilježavanja te znamenke je uvriježen u svim programskim jezicima kako bi se nula razlikovala od slova O. U radu s tastaturom treba imati na umu da za nulu i jedinicu (1) postoje posebne tipke jer je poznata činjenica da novi korisnici računala uporno na početku svog rada pritišću slova L i O navikli na rad s pisaćim strojem.

Tako čine greške koje računalo dojavljuje. Na posebne tipke za nulu i jedinicu brzo se navikavamo pa tako početne zabune nisu trajan problem pri programiranju.

Pojam broja u računalu obično se definira kao: decimalni broj koji se može kretati između minimalnog broja $1Ø^{-38}$ (2^{-127}) i maksimalnog broja $1Ø^{+38}$ (2^{+127}), a nula je također uključena.

U Commodoreu 64 te su granice određene od broja
1.7Ø141183E+38 do 2.93873588E—39, naravno i ovdje je nula uključena.

Brojevi se mogu prikazivati kao realni npr. 2.34 ili kao cjelobrojni (integeri) u granicama ±32767.

U realnim brojevima dvije su novosti, na koje većina ljudi nije navikla. Prvo, to što se decimalni brojevi NE ODVAJAJU DECIMALNIM ZAREZOM već DECIMALNOM TOČKOM. Zarez u računalu ima drugu namjenu — odvajanje dvaju brojeva. Druga činjenica je to što se mogu skraćeno pisati veliki brojevi ali tako kako većina vlasnika računala do sada nije susretala. Za oznaku eksponenta upotrebljava se veliko slovo E. Broj iza njega označuje eksponent broja 10, a njegov predznak nam kazuje da li je

skraćeno napisani broj velik (predznak +) ili malen (predznak —).

Primjer:

broj 155ØØØØØØØØØØØØØØØ

može se pisati kao:

1.55E+17

značenje toga broja je 1,55 × 10^{17} (jedno cijelo pedeset i pet puta deset na sedamnaestu).

Varijable

Varijable za razliku od konstanti mogu imati promjenljive vrijednosti. Općenito u BASIC-u ime varijable može biti slovo međunarodne abecede (A—Z) ili kombinacije slova A—Z i brojeva 0—9.

Na računalu Commodore postoje tri vrste varijabli: varijable realnih brojeva, cjelobrojne varijable i tzv. string-varijable.

Varijable kojima se daju vrijednosti realnih brojeva stručno se nazivaju varijable »Floating--Point« (varijable pomične decimalne točke).

Općenito imaju oblik XY gdje je X slovo međunarodne abecede (A—Z), a Y slovo (A—Z) ili broj (0—9).

Varijable realnih brojeva mogu sadržavati vrijednosti od ±1.7Ø141183E+38 do ±2.93873588 E—39.

Mogu imati oblike npr.:

RB=7559.2

Z=11.25E+8

C4=5.48E—6

A=76.83 itd.

Vidljivo je da ime varijable može biti i samo jedno slovo.

Druga vrsta varijabli jesu cjelobrojne (integer) varijable. Imaju oblik XY%, a za X i Y mogu biti upotrijebljena slova i brojevi kao što je rečeno za varijable realnih brojeva. Cjelobrojnim varijablama mogu biti pridružene brojevne vrijednosti u rasponu ±32767. Uz imena varijabli stoji znak % koji objašnjava računalu o kojem tipu varijabli je riječ.

Cjelobrojne varijable oblikom mogu biti:

D% = 42

Y% = —764

I3% = 19762

ZV% = —91

Dakle, ako želimo raditi s cjelobrojnim varijablama ne smije se izostaviti znak %.

Uz ove dvije vrste varijabli računalo C64 radi i s tzv. string-varijablama, odnosno varijablama u koje se može upisati neki tekst. Općenito oblik im izgleda: XY$.

Za X i Y vrijedi sve što je rečeno uz prije spomenute vrste varijabli. Znak $ objašnjava računalu da je to string-varijabla. Sadržaj string-varijable može imati proizvoljnu dužinu od 0 do 255 znakova.

String-varijable mogu imati oblik kao na primjerima:

C$ = »COMMODORE 64«

ZR$ = »SUMA JE«
P$ = »REZULTAT IZNOSI«
M2$ = »ZBROJI DVA BROJA«

Iz primjera se vidi da sadržaj string-varijabli mora biti dan u navodnicima, što treba imati na umu pri pisanju programa.

Treba spomenuti da imena varijabli mogu biti i duža od dva znaka (X i Y), ali računalo Commodore 64 može razlikovati samo prva dva znaka. Dakle. računalo neće razlikovati varijable nazvane ZAGREB i ZADAR jer su im prva dva znaka ista.

Upoznavši se sa sve tri vrste varijabli na Commodoreu 64 može se dati jednostavan primjer njihove zajedničke upotrebe.

Pretpostavimo da želimo od broja 56.5 oduzeti broj 11, te na ekranu dobiti izvještaj o rezultatu. Upotrijebit ćemo sve tri vrste varijabli — varijablu A kojoj dajemo vrijednost 56.5, cjelobrojnu varijablu B% s vrijednošću 11 i string-varijablu C$ čiji je sadržaj »REZULTAT JE«.

Kako bi taj kratki program izgledao može se vidjeti na primjeru:

```
15 A = 56.5
25 B% = 11
35 C$ = "REZULTAT JE"
45 PRINT "A = "; A, "B% = "; B%
55 PRINT C$; A — B%
```

Program se utipka u računalo (nakon svake programske linije treba pritisnuti tipku RETURN) te ga se može pokrenuti naredbom RUN. Na ekranu se dobiva:

```
RUN
A = 56.5 B% = 11
REZULTAT JE 45.5
READY
```

READY označava da je računalo spremno za daljnji rad.

U računalu C64 postoje još i tri posebne varijable zapisane u njegovoj »pameti«. To su varijable ST, TI i TI$, a tim imenima korisnici ne smiju nazivati svoje varijable.

Varijabla ST je tzv. status-varijabla i ona sadržava prikladnu vrijednost prije izvedene operacije isključivanja i uključivanja.

Varijabla ST mijenja svoju vrijednost (Ø) onda kada se javlja greška, odnosno problem pri upisivanju ili učitavanju podataka i programa na ili sa kazetofonske vrpce ili diskete. Ako su podaci ili programi krivo spremljeni ili učitani vrijednost varijable ST će biti različita od nule. Pravilno rukovanje podacima ili programima, dakle, može se ispitati pozivajući sadržaj varijable ST pomoću naredbe PRINT ST.

TI je varijabla koja ima značenje u vezi sa satom »ugrađenim« u računalo C64 (interni clock). Taj interni sat starta po uključenju računala, a rad mu se prekida samo pri radu s vanjskim jedinicama. Dakle, sadržaj varijable TI zapravo je vrijeme koje je proteklo u radu s računalom. Varijabla TI ima vrijednost Ø pri

uključenju računala, a vrijednost joj se može promijeniti, »sat vratiti unazad« kada promijenimo sadržaj varijable TI$.

TI$ je string-varijabla dužine 6 znakova, odnosno znamenki. Zapravo je mjesto za upis vremena (sata, minuta i sekundi) kada želimo da naš Commodore 64 radi kao sat. U string se vrijeme upisuje ovako:

Sati se upisuju na prva dva mjesta u stringu, minute na srednja mjesta, a sekunde na dva posljednja mjesta stringa. Dakle, string može izgledati npr. ovako:

TI$ = "111525"

a to znači 11 sati, 15 minuta i 25 sekundi.

Ako se računalo isključi, željeni string treba ponovno odrediti — definirati.

Malo vježbe i varijable neće praviti poteškoća. Njihova upotreba ne zaobilazi niti jedan program — od matematičkih pa sve do oblikovanja tablica i polja podataka.

Za oblikovanje polja na računalu C64 služimo se s tri tipa oblikovanja:

dimenzija 1 npr. oblika XY(2)
dimenzija 2 npr. oblika XY(2,2)
dimenzija 3 npr. oblika XY(2,2,2).

Polja do 11 elemenata (indeks (Ø—1Ø)) mogu se postići bez dodatnih definicija. Ako su potrebna veća polja, treba ih posebno dimenzionirati instrukcijom DIM. Tada se dobiva oblik:

DIM A(X,Y,Z)

što odgovara matematičkom izrazu za (X+1)*(Y+1)*(Z+1) elemenata, a prvi element je A(Ø,Ø,Ø).

Imajući sve ovo u vidu, te uz pomoć poglavlja o instrukcijama BASIC-a moći ćete polako početi izrađivati vlastite programe. Sustavno i bez žurbe predeni sadržaji olakšat će vam savladavanje osnova BASIC-a.

Programiranje upotrebom instrukcija GOTO, IF . . . THEN, FOR . . . NEXT

Upotrebom ovih instrukcija ćemo pokazati kako se mogu izvesti prvi programski koraci u BASIC-u.

Instrukcija **GOTO** (idi na) izvršava bezuvjetan skok unutar programa. Na primjer u nekom programu:

```
.
.
145 GOTO 521
.
.
```

znači da će program po nailasku na liniju 145 preskočiti sve instrukcije i doći na adresu 521 (programsku liniju) otkuda će nastaviti izvršavanje programa.

Npr.

15 PRINT "UNESI 2 BROJA"

```
25 INPUT X, Y
35 PRINT "NJIHOV ZBROJ JE : ";X+Y
45 GOTO 15
55 END
```

U ovom programu računalo će izračunati zbroj dvaju brojeva i kada ispiše rezultat (linija 35) prelazi na liniju 45 koja vraća program na liniju 15. Tom linijom program počinje ispočetka. Tako računalo nikada neće stići do linije 55 END. Program se može prekinuti tipkom RUN/STOP.

Isto tako kao primjer može poslužiti:

```
15 PRINT "C64"
25 GOTO 15
```

računalo će sada neprekidno prikazivati:

```
C64
C64
C64
.
.
C64
```

sve dok ne pritisnemo tipku RUN/STOP.

IF . . . THEN znači ako . . . tada. Ovim instrukcijama je omogućen uvjetan skok, npr.:

```
35 IF X=Y THEN 66
```

To znači da će računalo preskočiti tek pri ispunjenom uvjetu X=Y. Ako je X=Y program skače na liniju 66. Ako taj uvjet nije ispunjen računalo ovu liniju preskače i ide na slijedeću. Program može izgledati npr.:

```
15 REM zbrajanje i oduzimanje
25 PRINT "DA LI ZELITE : "
35 PRINT "1-ZBRAJATI ILI"
45 PRINT "2-ODUZIMATI"
55 PRINT "DAJTE ODGOVOR S 1 ILI 2"
65 INPUT A
75 IF A=1 THEN 1Ø5
85 IF A=2 THEN 155
95 GOTO 55
1Ø5 REM zbrajanje
115 PRINT "UNESITE 2 BROJA"
125 INPUT X, Y
135 PRINT "ZBROJ = ";X+Y
145 GOTO 15
155 REM oduzimanje
165 PRINT "UNESITE 2 BROJA"
175 INPUT X, Y
185 PRINT "RAZLIKA = ";X—Y
195 GOTO 15
215 END
```

Pomoću linija 75 i 85 računalo cjelovito upravlja programom. Ako bismo nakon linije 65 unijeli npr. broj 3 dobili bismo ponovni upit računala. Pokušajte utipkati program i komunicirati s računalom. Nakon svakog proračuna program se instrukcijom GOTO vraća na početak.

Pogledajmo drugi primjer:

```
15 A=Ø
25 PRINT "C64"
35 A=A+1
45 IF A<5 THEN 25
55 END
```

Po izvršenju dobivamo:

```
RUN
C64
C64
C64
C64
C64
```

Dakle, točno pet puta će se željeni "C64" prikazati na ekranu. Dok se ne prikaže peti puta program se linijom 45 vraća na početak. Po petom prikazu izvršenje se prekida.

FOR . . . NEXT- u prethodnom primjeru je instrukcija GOTO uvijek vraćala program na početak. Ako unaprijed znamo koliko puta treba izvršiti program, onda umjesto GOTO možemo dati instrukcije FOR i NEXT. Ako je u program upisana jedna instrukcija onda i druga mora biti upisana.

Npr.:

```
15 FOR A=1 TO 4
25 PRINT "C64"
35 NEXT A
```

po izvršenju imamo:

```
RUN
C64
C64
C64
C64
```

Varijabla u obje instrukcije mora biti ista. FOR i NEXT tvore petlju. U ovom primjeru linija 15 ima značenje: Neka A ide od 1 do 4, a instrukcija znači — uzmi slijedeći A.

Uz ove instrukcije je moguće primijeniti i instrukciju STEP.

Primjer:

```
15 FOR X=1 to 1Ø STEP .5
25 PRINT X,
35 NEXT X
```

prikaz će biti:

```
RUN
1       1.5     2       2.5
3       3.5     4       4.5
5       5.5     6       6.5
7       7.5     8       8.5
9       9.5     1Ø
```

To znači da X ide od 1 do 10 s korakom od 0,5 (u programu se ne mora pisati nula — dovoljna je samo decimalna točka — u ovom primjeru (.5), a linija 35 znači uzmi slijedeći X. Pokušajte izvesti program za obratni primjer da se ide od 10 prema 1.

Upotreba slučajnih brojeva — instrukcija RND

Često u programu trebamo generator slučajnih brojeva, ili za igru simulaciju bacanja kocke. To možemo postići ovako:

```
15 PRINT INT(6*RND(1))
```

po izvršavanju imamo npr.:

```
RUN
1       2       4       6
```

3 3 2 5
4 5

Instrukcija INT određuje cjelobrojnu vrijednost pri generiranju slučajnih brojeva instrukcijom RND. Ako bi izostala instrukcija INT, u prikazu bismo dobili neke decimalne vrijednosti npr.:

```
15 PRINT 6*RND(1)
```

prikaz:

```
RUN
4.9718388
3.2964321 itd.
```

Ako želimo ograničiti broj dobivenih vrijednosti u prikazu može se primijeniti ova rutina:

```
15 FOR X=1 TO 3
25 PRINT 6*RND(1),
35 NEXT
```

Broj prikazanih vrijednosti ovisi o liniji 15 iz koje se može vidjeti da bismo startanjem ove rutine dobili prikaz tri vrijednosti.

Upotrebom ovih zakonitosti i dobrim svladavanjem ostalih instrukcija i njihovih primjena uspjet ćete izvesti vaše prve programe — igre, proračune ili što drugo.

OPERACIJE U BASIC-u

Operacije u BASIC-u pa tako i na računalu Commodore 64 dijelimo na računske (aritmetičke) operacije, poredbene operacije i logičke operacije.

U aritmetičke operacije pripada:

zbrajanje (+)
oduzimanje (—)
množenje (*)
dijeljenje (/)
potenciranje (↑).

Poredbene operacije sadržavaju pojmove:

< > (različito)
> (veće od)
< (manje od)
> = (veće ili jednako)
< = (manje ili jednako)

Poredbene se operacije upotrebljavaju za određivanje logičkih odnosa među izrazima.

Znak jednakosti (=) upotrebljava se i u aritmetičkim i u poredbenim operacijama.

Taj znak pridružuje aritmetičku ili logičku veličinu nekoj varijabli. Ako ga u programu upotrebljavamo npr. kao:

A = A+5,

to znači da je nova vrijednost varijable A jednaka staroj vrijednosti varijable A uvećanoj za 5.

Uz aritmetičke i poredbene operacije postoje i logičke operacije sa svojim operatorima:

NOT (logičko NE)
AND (logičko I)
OR (logičko ILI).

Aritmetičke operacije

Najjednostavnije ih je objasniti promatrajući ih izvan programskih linija kada računalo C64 radi kao kalkulator.

Prvo treba vidjeti što se dobiva kada utipkamo: PRINT "C64" i pritisnemo RETURN.

Dobit ćemo izgled ekrana:

PRINT "C64"
C64
READY
■

Obrišimo ekran s SHIFT i CLR/HOME te pokušajmo sada to s matematičkim operacijama. Utipkajte:

PRINT 17 + 41

pritisnite RETURN i dobivate rezultat

58

Obrišite ekran (SHIFT i CLR/HOME) pa tako nešto pokušajte s drugim matematičkim operacijama npr.:

PRINT 16 — 4.85

zatim RETURN i dobivate rezultat

11.15

Uvidjet ćete da nema nikakvih problema. Razlika između aritmetike na računalu i »papirne« matematike je samo u novim oznakama — simbolima operatora za množenje, dijeljenje i potenciranje. U školi smo naučili da se za znak množenja upotrebljava točka ili križić (2·2, 4×5), a ovdje umjesto toga postoji zvjezdica (✳). U dijeljenju umjesto dvotočke na računalima postoji kosa crta (/). Pogledajmo množenje i dijeljenje na primjerima:

PRINT 4✳3 (RETURN) i kao rezultat izlazi
12

ili

PRINT 15/5 (RETURN) i rezultat izlazi
3

Da podsjetimo, ovdje smo još naznačili nužnost pritiska na tipku RETURN, kao znak da je unošenje podataka završeno. U daljnjem tekstu tog podsjetnika više nema, ali zapamtimo: svako unošenje zahtijeva na kraju RETURN.
U dijeljenju treba reći da kosa crta nikako nema značenje kose zagrade već se upotrebljava isključivo kao znak za dijeljenje.

Znak za potenciranje je uspravna strelica (↑). Na nekim računalima je taj simbol dan kao dvije zvjezdice (✳✳). Poznavanjem i tog simbola lakše ćete se snaći pri preradi programa BASIC s drugih računala.

Potenciranje je matematička operacija u kojoj se izračunava potencija nekog broja npr. osma potencija broja dva. To je isto kao kada bismo uzastopno pomnožili broj 2 osam puta. Dakle, i tako možemo računati: Potrebno je izračunati 2^8.

Prvi put je dug i nepraktičan:

PRINT 2✳2✳2✳2✳2✳2✳2✳2

a rezultat je

256

Mnogo je praktičnije to izračunati operacijom potenciranja:

PRINT 2↑8

256

Iz svih primjera se vidi da je na početku stajala naredba PRINT. Međutim kada tako radimo s računalom može se naredba PRINT zamijeniti upitnikom (?), odnosno kao na primjeru:

? 172 — 41

131

Dakle, upitnikom (?) skraćujemo posao jer utipkavanje naredbe PRINT postaje nepotrebno.

Složene matematičke operacije

Svaki ozbiljniji matematički izraz u sebi sadržava različite matematičke operacije. Prvo treba znati — kako da izraz prilagodimo za upis u računalo, bilo za direktno izračunavanje ili unošenje u programske linije.

Pogledajmo kako bismo u BASIC-u napisali jednadžbu pravca:

$$Y - Y1 = \frac{Y2 - Y1}{X2 - X1}(X - X1)$$

U BASIC-u nisu moguća izjednačenja dvaju izraza pa je Y1 potrebno prebaciti na desnu stranu jednadžbe:

$$Y = \frac{Y2 - Y1}{X2 - X1}(X - X1) + Y1$$

Tek sada se jednadžba pravca može napisati u BASIC-u kao:

Y = (Y2 — Y1)/(X2 — X1)*(X — X1) + Y1.

Tu se srećemo s novim pojmom — prioritetom aritmetičkih operacija, odnosno koja će se izvršiti prva, a koja posljednja.

Pogledajmo primjer:

12 + 12 / 4

rezultat bi mogao biti 6 ili 15, ovisno o tome koju bismo operaciju prvu izvršili. Računalo C64 daje kao rezultat 15. Sve ovisi upravo o prioritetu aritmetičkih operacija, a u Commodoreu 64 to izgleda ovako:

1. stupanj — znak minusa ALI SAMO KAO PREDZNAK BROJA
2. stupanj ↑ potenciranje
3. stupanj */ množenje i dijeljenje
4. stupanj +— zbrajanje i oduzimanje

Prema ovom stupnjevanju prioriteta izraz

? 7*12—5 daje rezultat

79

Vidljivo je da će računalo prvo obavljati operacije višeg stupnja pa zatim sve operacije prema prikazanom prioritetu. To treba imati na umu pri definiranju matematičkih izraza.

Isto tako može se na slijedećem primjeru vidjeti veza pravilnog oblikovanja instrukcija imajući u vidu područje varijabli i stupnjevanje matematičkih operacija.

Uzmimo kao primjer kvadratnu jednadžbu:

$$X_{1,2} = \frac{-B \pm \sqrt{B^2 - 4AC}}{2A}$$

Ova jednadžba se u BASIC-u mora dati kao dva izraza, i to zato što se dvije operacije ne smiju naći jedna do druge (ovdje se misli na + i — pred kvadratnim korijenom), a ni ime varijable ne smije biti oblika X1,2. Dakle potrebna su dva izraza:

```
X1 = (—B+(B↑2 — 4*A*C)↑(1/2))/(2*A)
X2 = (—B—(B↑2 — 4*A*C)↑(1/2))/(2*A)
```

Pravilo o prioritetima može se malo promijeniti uvođenjem okruglih zagrada. Naime, vratimo se na već spomenuti primjer. Rekli smo da:

```
? 7*12 — 5     daje rezultat
79
```

Ako sada upotrijebimo okruglu zagradu

```
? 7*(12 — 5)     dobit ćemo rezultat
49
```

Dakle računalo prvo rješava izraz u zagradi, a zatim prelazi na operacije prema prioritetu. Ovdje se može spomenuti da se za ljepši prikaz računanja na ekranu mogu upotrebljavati oblici, npr.:

```
? "9 / 3 = "; 9/3     pa će rezultat izgledati:
9/3 = 3
```

Logičke operacije (AND, OR, NOT)

AND-operacija (operacija I) služi za određivanje logičke veze između dva izraza ili dviju varijabli. Ako su oba izraza »istinita« i zaključak je »istinit«. Ako su oba izraza ili samo jedan »neistiniti« i zaključak nije »istinit«.

U BASIC-u logička jedinica ima obilježje istine, a logička nula neistine.

Što se matematičkih izraza tiče: logička jedinica (1) je bilo koji broj različit od nule, a logička nula (Ø) odgovara nuli.

AND-operacija može se prikazati i tablicom stanja:

TABLICA STANJA ZA **AND**-OPERACIJU:

izraz A	izraz B	A AND B
1	1	1
1	Ø	Ø
Ø	1	Ø
Ø	Ø	Ø

OR-operacija (operacija ILI) određuje logičku ovisnost dvaju izraza. Dovoljno je da su jedan ili drugi izraz istiniti (ili oba), pa da je i OR (ILI)-operacija istinita.

TABLICA STANJA ZA **OR**-OPERACIJU:

izraz A	izraz B	A OR B
1	1	1
1	Ø	1
Ø	1	1
Ø	Ø	Ø

NOT-operacija (operacija NE) služi za dobivanje obrnutih vrijednosti (komplemenata) izraza ili varijabli.

Na primjer, ako je Z = Ø onda je NOT Z = 1. To znači, ako je npr. B bilo koja veličina različita od nule (Ø) da je tada NOT B = Ø.

Na kraju ovog upoznavanja s operacijama u BASIC-u možemo ih sve svrstati prema prioritetu odnosno redoslijedu izvršavanja. Potpuna slika prioriteta na računalu C64 izgleda ovako:

NAJVIŠI PRIORITET:

— znak minusa ALI SAMO KAO PREDZNAK BROJA
↑ POTENCIRANJE
NOT operacija
*/ MNOŽENJE I DIJELJENJE
+ — ZBRAJANJE I ODUZIMANJE
SVI POREDBENI OPERATORI
AND-operacija

NAJNIŽI PRIORITET:

OR-operacija

NAREDBE I FUNKCIJE

U ovom popisu nalaze se naredbe i funkcije koje čine COMMODORE 64 BASIC. Uza svaku nalazi se kraći opis njene funkcije i način sintakse (kako se ispravno piše). Najbolje ćete se naučiti služiti računalom tako da upisujete te naredbe i primjere za njih, te ih mijenjate i pratite rezultate na ekranu. Ako i pogriješite nešto, kompjuteru ne možete time naškoditi.

CONT (continue)

— ovom komandom nastavlja se izvršavanje programa koji je zaustavljen naredbom STOP ili END ili pritiskom na tipku RUN/STOP. Program nastavlja rad od linije na kojoj je zaustavljen. No, nastavka neće biti ako ste mijenjali (editirali) program ili ako je zaustavljanje uzrokovano greškom u programu. Tada će se pojaviti poruka CAN'T CONTINUE ERROR.

LIST

— naredba LIST služi za ispisivanje programskih linija na ekranu. Ovisno o formatu naredbe, moguće je izlistati cijeli program ili samo dio programa.

LIST	izlistava cijeli program
LIST 1Ø—	izlistava program od linije 10 nadalje do kraja
LIST 1Ø	izlistava samo liniju 1Ø
LIST —1Ø	program se izlistava do linije 1Ø
LIST 1Ø—5Ø	program se izlistava od linije 1Ø do linije 5Ø, uključujući i te dvije granične

LOAD

— ovom naredbom učitava se program s kazetofona ili disk-jedinice u računalo. Iza nared-

be može stajati ime programa, broj jedinice s koje se učitava (1-kazetofon, 8-disk) i sekundarna adresa (Ø ili 1). Ako je sekundarna adresa jednaka 1, tada će se program učitati na istu memorijsku lokaciju s koje je spremljen. U protivnom, programi koji se učitavaju samo s naredbom LOAD, učitavaju se od adrese 2048 nadalje. Pri učitavanju na ekranu se pojavi ime programa i riječ 'FOUND'. Tada treba pritisnuti razmaknicu, CTRL, COMMODORE ili tipku ←, da bi se program učitao. Ako se program pravilno učita, na ekranu se pojavljuje riječ 'READY'.

Sintaksa:

LOAD	učitava prvi program sa kazete
LOAD"TEH1"	učitava program s imenom "TEH1" sa kazete
LOAD A$	traži i učitava program čije se ime nalazi u varijabli A$
LOAD"TEH1",8	traži i učitava program "TEH1" s diska
LOAD"*",8	učitava prvi program s diska
LOAD" ",1,1	učitava prvi program s vrpce na isto mjesto u memoriji s kojeg je spremljen

NEW

— upotrebljava se da bi se izbrisao program koji je trenutačno u memoriji i čisti sve varijable. Ta naredba može se dati direktno ili se može nalaziti unutar programa. Tada treba biti pažljiv da se slučajno ne bi izbrisao višesatni posao utipkavanja programa.

RUN

— ovom naredbom starta se program koji se trenutačno nalazi u memoriji kompjutera. Time je uzrokovano i automatsko čišćenje varijabli komandom CLR. Ako želite upotrijebiti varijable koje su već postavljene prijašnjim izvođenjem programa, tada ga morate startati naredbom CONT ili GOTO.

RUN	izvršava program od prve linije (i resetira varijable)
RUN 45Ø	starta program od linije 45Ø
RUN X	daje poruku UNDEFINED STATEMENT ERROR. Varijable nisu dopuštene u linijskim brojevima.

SAVE

— naredbom SAVE sprema se program koji se trenutačno nalazi u memoriji računala na kazetu ili disketu. Iako to nije nužno, preporučljivo je iza naredbe SAVE svakom programu dati ime, radi lakše identifikacije pri učitavanju. Iza imena može stajati adresa uređaja na koji se program sprema (1-kazetofon, 8-disk jedinica), te sekundarna adresa, koja može iznositi 1-oznaka za punjenje na istu adresu, 2-oznaka kraja vrpce i 3-kombinacija obje ove funkcije. Pri spremanju programa na disk mora se navesti ime programa.

SAVE	sprema program bez imena na kazetu
SAVE"BETA"	sprema program s imenom "BETA" na kazetu
SAVE A$	sprema program na vrpcu, s imenom sadržanim u varijabli A$
SAVE"BETA",8	sprema program "BETA" na disk

VERIFY

— ovom naredbom može se usporediti program spremljen na vrpci ili disku s programom u memoriji kompjutera. Verify se obično upotrebljava nakon izvršenja naredbe SAVE, da bi se ispitalo da li je program korektno spremljen na vrpcu ili disk. Pri ispitu za verifikaciju programa s diska moramo upisati i ime, dok za program sa kazete to nije nužno. Ako se u toku usporedbe nađe neka greška, kompjuter to javlja porukom ?VERIFY ERROR.

VERIFY	provjerava prvi program s vrpce
VERIFY"BETA",8	traži program "BETA" na disku i uspoređuje ga s programom u memoriji

CLOSE

— naredba zatvara datoteku ili kanal prema uređaju. Broj datoteke treba biti isti kao u naredbi OPEN kojom je otvorena.

CLOSE 1 ili CLOSE X ili CLOSE 9*(1-K) itd.

CLR

— ovom naredbom brišu se sve varijable u memoriji, a program ostaje netaknut. Isti efekt nastaje izvođenjem naredbe RUN (prije izvođenja samog programa resetiraju se varijable).

```
10 X=25
20 CLR
30 PRINT X
```

CMD

— ovom naredbom šalju se podaci, koji se obično pokazuju na ekranu, na neki drugi uređaj. To može biti pisač ili datoteka na disku ili vrpci. Sam uređaj ili datoteka prije toga moraju biti »otvoreni« naredbom OPEN. Naredbu CMD treba slijediti broj ili brojčana varijabla.

Primjer:

OPEN 1,4	otvara uređaj 4, pisač
CMD 1	izlaz usmjeren na pisač
LIST	program se izlistava na pisaču, umjesto na ekranu

— da bismo podatke slali ponovno na ekran, trebamo poslati praznu liniju CMD uređaju pomoću PRINT#1, i zatim zatvoriti kanal naredbom CLOSE 1.

DATA

— DATA-linije služe za spremanje podataka unutar programa. Program učitava te podatke pomoću naredbe READ. Same linije DATA nemaju nikakvu izvršnu funkciju, pa ih možemo

smjestiti gdje nam najbolje odgovara. Podaci unutar DATA mogu biti numerički ili tekstualni, a tretiraju se kao kontinuirana lista, te se iščitavaju slijeva nadesno, od niže linije prema višima.

```
45 DATA 23,36,6,"zdravo",45,6,8,"itd"
```

DEF FN

— postavljanje korisničke funkcije koja se definira pod imenom i kasnije se može pozvati u programu. Takav rad štedi memoriju na onim mjestima gdje treba više puta primijeniti istu kompleksnu formulu. Formula se definira samo jedanput i kasnije se samo poziva ime funkcije. Ime slijedi iza FN, a sastoji se od 1 ili 2 znaka, od kojih prvi mora biti slovo, a drugi slovo ili broj.

Primjer:

```
1Ø DEF FNAB(X)=23*(11.234-X/Ø.45)
2Ø PRINT FNAB(9)
```

DIM

— ovom naredbom dimenzioniraju se polja podataka. Najniži broj mjesta u polju je Ø, a najviši 32767. Polja mogu biti jednodimenzionalno, dvodimenzionalno, trodimenzionalno i više dimenzionalna, no najčešće služe prve tri mogućnosti. Samo polje može se sastojati od brojeva (realnih ili cijelih) ili od stringova. Za cjelobrojne varijable upotrebljava se znak % iza imena varijable, a za stringove $ iza imena. Naredba za dimenzioniranje izvršava se samo jednom u toku programa i zato je dobro da je pozicioniramo na početku.

Primjeri:

```
1Ø DIM C$(35),G6(17),CD%(3,5,3)
         .        .       .
         .        .       .
         .        .       .
       string   realno  integer
       polje    polje   polje
```

ili

```
1Ø DIM A(1ØØ)
2Ø DIM Z(8,4),Y(5,4,2)
3Ø DIM X7%(456)
4Ø DIM ST$(23,8)
```

END

— pri nailasku na ovu naredbu kompjuter zaustavlja rad i ispisuje poruku READY. Izvršavanje se može nastaviti naredbom CONT.

FOR . . . TO . . . STEP

— zajedno s naredbom NEXT, ova naredba tvori zatvorenu petlju koja će se izvršiti onoliko puta koliko je definirano u varijabli iza FOR, ili ako upotrijebimo STEP, tada će se izvršiti n/STEP puta. Ako ne postoji dio STEP, tada je korak povećavanja=1.

Primjer:

```
1Ø FOR F=1 TO 25
2Ø PRINT F
3Ø NEXT F
```

ili
```
10 FOR K=100 TO 0 STEP -2
```
ili
```
10 FOR K=PI TO 7*PI STEP 0.02
```

GET

— naredbom GET čitaju se podaci s tastature, i to svaki put po jedan znak. Ako GET očekuje numeričke podatke, a tipkamo neki drugi znak, dobit ćemo na ekranu ?SYNTAX ERROR. Zbog toga je bolje tipke čitati kao stringove i zatim ih konvertirati u brojeve.

Primjer:
```
10 GET A$:IF A$="" THEN 10
10 GET A$,B$,C$,D$,E$ — čita pet tipki
```

GET#

— radi isto kao i GET, ali se ne čita tastatura već znakovi dolaze s drugog uređaja.

Primjer:
```
 5 GET#1,A$
10 OPEN 1,3:GET#1,BF$
```

GOSUB

— ovom naredbom program se šalje na neki potprogram koji se izvrši i zatim se vraća na slijedeću liniju. Potprogram obavezno na kraju mora imati naredbu RETURN, koja označava njegov kraj. Upotreba subrutina je vrlo korisna, jer ne moramo istu sekciju pisati više puta, te se štedi memorija.

Primjer:
```
10 GOSUB 500
 . . . . . .
 . . . . . .
500 FOR I=1 TO 400:PRINT " ":NEXT
505 RETURN
```

GOTO ili GO TO

— program skače na liniju čiji se broj nalazi iza naredbe GOTO i nastavlja izvršavanje od te linije. Ako ne postoji linija s tim brojem, kompjuter se zaustavlja s izvještajem o greški.

IF . . . THEN

— ovom naredbom kompjuter može analizirati određenu situaciju. Računalo bira između dvije mogućnosti, ovisno o rezultatu analize. Ako je tvrdnja nakon IF istinita (ispunjena), tada se izvršava dio linije iza THEN. Tu može stajati bilo koja BASIC-naredba. U suprotnome THEN se ignorira i program se nastavlja na prvoj slijedećoj liniji. Najčešće se u tvrdnji (uvjetu) iza IF primjenjuju logički operatori.

Primjeri:
```
100 IF N<=4 THEN X=X-1
```
ili
```
150 IF N>0 GOTO 350
```

INPUT

— naredba INPUT očekuje od korisnika programa da upiše podatke koji su potrebni za daljnju obradu. Iza naredbe može slijediti i tekst u

navodnicima koji objašnjava korisniku što se traži od njega. Na ekranu se pojavljuje taj tekst i upitnik, iza kojeg se upisuju podaci. Iza INPUT i teksta stoji znak ;- a iza njega se nabrajaju varijable (obične ili string-$) u koje se podaci pohranjuju. U INPUT se može upisati i više od jednoga podatka, no tada podaci moraju biti odvojeni zarezom. Ako upišemo premalo podataka, na ekranu će se pojaviti dva upitnika (??). Ako se upiše višak podataka, pojavit će se poruka ?EXTRA IGNORED.

Primjeri:

```
100 INPUT A
150 INPUT B,C,D$
160 INPUT "IME I BROJ";I$,B
```

INPUT#

— ima istu funkciju kao i INPUT, s time da se podaci uzimaju iz prije otvorene datoteke s naredbom OPEN.

LET

— upotrebljava se da bi se nekoj varijabli pridijelila vrijednost. No u BASIC-u C64 to uopće nije potrebno, pa se LET niti ne rabi, već je tu zbog određene standardnosti jezika.

Primjer:

LET D=12 je isto kao D=12

NEXT

— NEXT se uvijek upotrebljava u sprezi s naredbom FOR...TO. Kad program naiđe na NEXT u petlji, ispita da li je postignuta gornja vrijednost kontrolne varijable. Ako granica nije dosegnuta, brojaču se dodaje 1 ili vrijednost definirana u naredbi STEP. Ako je granica dosegnuta, program nastavlja s izvođenjem prve naredbe iza NEXT.

Ako se jedna petlja nalazi unutar druge, treba paziti da se ne sijeku. Može se ići do 9 nivoa petlji, jedne unutar drugih.

Primjer:

```
10 FOR J=1 TO 5:FOR K=5 TO 15:FOR
   L=1 TO 10
20 NEXT L,K,J
```

ili

```
10 FOR A=1 TO 10
20 FOR B=1 TO 10
30 NEXT B
40 NEXT A
```

ili

```
10 FOR X=1 TO 235:NEXT
```

ON

— naredba ON se upotrebljava da bi se izračunao skok GOTO ili skok GOSUB. Ovisno o vrijednosti varijable iza ON-(1,2,3 itd.), program će skakati na liniju (prvu, drugu, treću itd.) definiranu iza GOTO (ili GOSUB). Ako se kao varijabla pojavi negativna vrijednost, dobivamo poruku ?ILLEGAL QUANTITY. U slučaju nule ili broja većega od broja linija iza GOTO, računalo ignorira naredbu taj put i nastavlja izvođenje na slijedećoj liniji. Naredba ON GOTO

može zamijeniti veći broj naredbi IF, te može znatno olakšati programiranje.

Primjer:

```
10 INPUT X:ON X GOTO 10,20,30,40,50
ili
ON X GOSUB 9000,230,9000
ON VS/2+1 GOSUB 90,10,350
```

OPEN

— naredba OPEN otvara kanal za ulaz i/ili izlaz na periferni uređaj (kazetofon, disk, pisač, ekran). Iza OPEN prvo dolazi logički broj niza (FILE-a) podataka, koji ga povezuje s naredbama CLOSE, CMD, GET#, INPUT#, PRINT#, a može imati vrijednost od 1 do 255. Zatim dolazi broj koji nam pokazuje na kojem uređaju podaci postoje. To su:

Ø tastatura
1 kazetofon
3 ekran
4 ili 5 pisač
8—15 disk

Iza toga dolazi sekundarna adresa — za kazetofon Ø znači čitanje, 1 — upisivanje i 2 — upisivanje s markerom za kraj vrpce. Za disk sek. adresa mora postojati, a na raspolaganju su vrijednosti 2—14, dok ostali brojevi služe unutar sistema DOS. Ime može biti dugo od 1 do 16 znakova.

Sintaksa naredbe:

OPEN /broj filea//uređaj//sek. adresa//"ime, /tip,/mod"/

primjeri:

OPEN 1,Ø	otvara tastaturu
OPEN 2,1,Ø,"D"	otvara kazetofon za čitanje, traži se file D
OPEN 9,3	izlaz na ekran
OPEN 3,4	otvara pisač
OPEN 1,4,Ø,"STRING"	šalje velika slova / grafiku na pisač
OPEN 1,4,7,"STRING"	šalje velika/mala slova na pisač

POKE

— naredba se sastoji od dva broja ili formule. Prvi broj označava adresu u memoriji u koju se pohranjuje vrijednost drugog broja (Ø—255). POKE služi za spremanje podataka, kontrolu grafike i zvuka, punjenje strojnih rutina itd.

Primjer:

```
POKE 36879,8
POKE A,B
```

PRINT

— to je naredba koju treba prvu zapamtiti, jer se pomoću nje na ekranu ispisuju podaci. Iza naredbe PRINT može stajati tekst u navodnicima, imena varijabli, funkcije, kontrolni karakteri. O znacima poput zareza (,), točke-zareza (;) ovisi da li će se tekst i podaci pisati s većim razmakom (,) ili odmah jedan iza drugoga (;).

Primjer:

```
PRINT"HALO"
PRINT"HALO",A$,B
```

```
PRINTJ:K:M
PRINT D+C
```

PRINT#

— služi za upisivanje podataka u logički niz. Mora se upotrijebiti isti broj s kojim je niz otvoren (OPEN). Izlaz ide na uređaj određen u naredbi OPEN. Zarez i točka-zarez imaju isti efekt kao u normalnoj naredbi PRINT.

Primjer:

```
100 PRINT# 1,"PODACI:":A.C%.D$
```

READ

— služi za čitanje podataka iz DATA-linija. Tipovi podataka moraju odgovarati tipovima varijabli u koje se upisuju.

Primjer:

```
 10 READ A,B,N$
 20 DATA 6,5,KNJIGA
100 FOR X=1 TO 5:READ A(X):NEXT
110 DATA 4.5,3,45,23.3
200 READ IME$,GRAD$,BROJ
210 DATA PERO,ZAGREB,41000
```

REM

— iza naredbe REM može stajati bilo kakav tekst. npr. objašnjenje linije. Kompjuter se na linije REM ne osvrće.

RESTORE

— resetira interni brojač podataka iz DATA-linija, i postavlja ga na prvu DATA-liniju. Tako se podaci mogu višestruko upotrebljavati.

RETURN

— naredba se mora nalaziti na kraju potprograma (subrutine) koji je pozvan naredbom GOSUB. Poslije naredbe RETURN program skače na prvu liniju ili naredbu iza naredbe GOSUB.

STOP

— naredba zaustavlja program, a na ekranu se pojavljuje poruka BREAK IN xxx (prekid u liniji). Program se može nastaviti izvoditi naredbom CONT.

SYS

— ovom naredbom. iza koje stoji adresa (Ø—65535) starta se program u strojnom jeziku, koji počinje na toj adresi.

WAIT

— naredba uzrokuje zaustavljanje programa sve dok se na određenoj adresi ne pojavi određena vrijednost. Iza naredbe slijedi adresa, te jedna ili dvije varijable. Format je: WAIT X,Y,Z. Sadržaj adrese se spaja s vrijednošću Y logičkom operacijom I (AND), a ako postoji Z, tada se rezultat prve operacije spaja sa operacijom EXILI (EX OR). Ovisno o rezultatu operacija (Ø ili 1), kompjuter čeka na promjenu sadržaja adrese ili nastavlja rad.

Numeričke funkcije

ABS(X)

— vraća apsolutnu vrijednost nekog broja, pa je rezultat uvijek pozitivan ili jednak nuli

ATN(X)

— vraća vrijednost arcus tangensa od X

COS(X)

— računa kosinus kuta X u radijanima.

EXP(X)

— daje X-tu potenciju matematičke konstante e (2.71827183).

FNxx(X)

— daje rezultat korisničke funkcije definirane u naredbi DEFFN.

INT(X)

— daje cjelobrojnu vrijednost X, bez decimala.

LOG(X)

— daje prirodni logaritam od X (po bazi e). Za preračunavanje u logaritam s bazom 1Ø, treba rezultat podijeliti s LOG(1Ø).

PEEK(X)

— vraća vrijednost koja se nalazi na adresi X (Ø—65535). Rezultat je cijeli broj između Ø i 255.

RND(X)

— RND kreira slučajni broj od Ø do 1. Kompjuter generira sekvenciju slučajnih brojeva zasnovanu računskom operacijom na startnom broju, zvanom sjeme. Taj broj nastaje pri uključenju računala. Ako je argument X pozitivan broj, generira se sekvencija pseudoslučajnih brojeva koja se može ponoviti ako se upiše isti broj-»sjeme«. Takva poznata sekvencija »slučajnih« brojeva može poslužiti pri testiranju programa. Ako je X jednak nuli, tada RND generira brojeve iz hardverskog »sata« kompjutera. Negativni argument X daje sa svakim pozivom funkcije novi startni broj.

Primjer:

PRINT INT(RND(Ø)*5Ø)
daje slučajne cijele brojeve između Ø i 49

PRINT INT(RND(1)*15Ø)+1ØØ
slučajni brojevi između 1ØØ—249

SGN(X)

— daje vrijednost ovisnu o predznaku X. Ako je X pozitivan dobivamo 1, za X=Ø dobijemo Ø i za negativan X dobivamo —1.

SIN(X)

— daje vrijednost sinusa kuta X u radijanima.

SQR(X)

— daje kvadratni korijen od X.

TAN(X)

— daje vrijednost tangensa kuta X u radijanima.

USR(X)

— pri upotrebi ove funkcije, program skače na strojni program, čija je adresa pohranjena u adresama 785 i 786. Ta adresa mora već biti pohranjena pomoću naredbi POKE. Argument X

predaje vrijednost strojnom programu, koji drugu vrijednost vraća programu BASIC.

ASC(X$)

— vraća ASCII kod prvoga znaka X$.

CHR$(X)

— reverzna ASC funkciji, ispisuje znak koji odgovara ASCII vrijednosti X.

LEFT$(X$,X)

— vraća dio stringa, točnije X lijevih slova stringa X$.

LEN(X$)

— vraća broj znakova sadržanih u stringu X$.

MID$(X$S,X)

— vraća dio stringa X$ od X znakova počevši od S-toga znaka.

RIGHT$(X$,X)

— vraća X desnih znakova stringa X$.

STR$(X)

— vraća string reprezentaciju numeričke vrijednosti argumenta.

VAL(X$)

— radi suprotno od STR$ funkcije, vraća numeričku vrijednost podataka u stringu. Ako prvi znak nije broj ili predznak vrijednosti, dobiva se nula.

primjer:

X=VAL("123.456")   X=123.456

X=VAL("34FG5H")   X=34

X=VAL("VHG65I")   X=Ø

FRE(X)

— vraća broj slobodnih bytova u memoriji, neovisno o X.

POS(X)

— vraća poziciju na ekranu (Ø—39) na kojoj se nalazi kursor.

SPC(X)

— služi u naredbi PRINT da se preskoči X znakova.

TAB(X)

— upotrebljava se s naredbom PRINT. Slijedeći znak se piše za X znakova udaljen od ruba.

Dodatne matematičke funkcije koje se dobivaju iz funkcija BASIC-a C64

SEKANS SEC(X)=1/COS(X)

KOSEKANS CSC(X)=1/SIN(X)

KOTANGENS COT(X)=1/TAN(X)

ARKUSSINUS ARSIN(X)=ATN(X/SQR(1-X↑2))

ARKUSKOSINUS ARCOS(X)=—ATN(X//SQR(1-X↑2))+Π/2

ARKUSKOTANGENS ARCOT(X)=ATN(X)+Π/2

SINUS HIPERBOLNI SINH(X)=(EXP(X)—EXP(-X))/2

KOSINUS HIPERB. COSH(X)=(EXP(X)+EXP(-X))/2

NEDOSTACI COMMODOREA 64 BASICA-a, MOGUĆNOSTI UPOTREBE DODATNOG SIMON'S BASIC-a

Svako kućno računalo uz svoje prednosti ima i poneku manu. Te mane nikada nisu takve da bi korisnike računala morale obeshrabrivati, tim više što se u pogledu računala mora uvijek gledati relacija cijena/mogućnosti.

Računalo Commodore 64 je vrlo dobro sklopovski koncipirano, postoji ogromna programska podrška raznih namjena, mogu se dokupljivati razni sklopovski dodaci, a ipak to računalo ima jednu manu — relativno loš i skroman BASIC.

Razlozi takvog skromnog i relativno lošeg BASIC-a mogu se naći promatrajući cjelokupni razvoj tvrtke Commodore i cijele kompjuterske tehnologije uopće.

Računala su sve do nekoliko godina unazad bila privilegija uskoga kruga korisnika, vještih programera, studenata, znanstvenika, itd., pa ni programski jezik poput BASIC-a nije morao biti baš krajnje jednostavan i dostupan. Tvrtka Commodore na računalu C64 ima varijantu BASIC-a koja zadovoljava sve do upotrebe grafičkih i glazbenih mogućnosti, a upravo one potiču najkreativniju primjenu računala te usavršavanje programiranja.

Ako se grafičke i glazbene mogućnosti žele potpuno iskoristiti, tada treba posegnuti za instrukcijama PEEK i POKE, odnosno direktno pristupati određenim memorijskim lokacijama. Postoji i druga znatno kompliciranija mogućnost — napuštanje razumljivog BASIC-a i upotreba složenijeg strojnoga koda (machine code). Naravno da ove činjenice nimalo ne oduševljavaju početnike, pa čak ni znalce u programiranju naviknute na udobnije varijante višeg programskog jezika BASIC.

Ova skromnija varijanta BASIC-a proizašla je iz pomalo zastarjele logike kompjuterskih tvrtki — da će stroj svladati u potpunosti svi oni kojima je potreban. Masovnom upotrebom računala ta logika nije više ispravna, tako da se danas pronalaze i uvode što jednostavnije i bolje varijante — dijalekti jezika BASIC. Takvu težnju vidimo i na najnovijim računalima tvrtke Commodore — bolji i ekonomičniji BASIC, lakši za uporabu širokom krugu korisnika, ali svejedno opet ponešto manjkav prema BASIC-u na računalima drugih tvrtki.

Uvidjevši da bi ograničeni BASIC mogao uzrokovati mnoge poteškoće korisnicima Commodorea 64, javio se na tržištu, gotovo paralelno s izlaskom računala, program Simon's BASIC. Taj program omogućuje proširenje osnovnog BASIC-a Commodorea 64 za stotinjak novih naredbi.

Uz ovaj pojavio se i Simon's BASIC II, ali u sebi ima i neke pogreške, tako da se većina korisnika zadržava na osnovnoj varijanti Simon's BASIC-a.

Naredbe Simon's BASIC-a tako su složene da znatno olakšavaju uporabu računala i programiranje, a time povećavaju i njegovu iskoristivost.

U nas se Simon's BASIC nalazi najčešće kao program snimljen na kazeti, što znači da je u masovnoj upotrebi i zaista su rijetki vlasnici računala Commodore 64 koji ga ne upotrebljavaju. Naš je savjet da taj program svakako nabavite i da ga što više upotrebljavate jer daje novu dimenziju iskorištenja računala C64.

Osim na kazeti Simon's BASIC može se nabaviti zapisan na disketi ili u dodatnom ROM kertridžu (memorijskom pakovanju). Snimljen na disketi rijetko se susreće u nas iz jednostavnog razloga što disk-jedinice ima vrlo malo vlasnika.

Upisan u memoriju ROM najpraktičnije je rješenje za upotrebu jer je njegovo učitavanje pri radu trenutačno.

Takvo dodatno memorijsko pakovanje priključuje se na stražnjoj strani računala.

Iz narednog teksta moći će se uočiti da programiranje na računalu Commodore 64 nije veći problem ako se upotrebljava Simon's BASIC.

INSTRUKCIJE U SIMON'S BASICU

Simon's BASIC je program koji uveliko poboljšava mogućnosti Commodorea 64. Njegovim aktiviranjem dobivate više od stotinu novih naredbi, koje obogaćuju prilično siromašan BASIC C-64. Pokrivena su razna područja programiranja i to: olakšice pri programiranju, zaštita programa, strukturirano programiranje, varijable, računanje, operacije s nizovima, formiranje izlaza, kontrola ulaza i izlaza, grafika, znakovi, spriteovi, zvuk i periferni uređaji.

Kako se upute za primjenu ovog vrlo korisnog programa u nas dosta teško nalaze, ovdje ćemo prikazati naredbe i njihovu sintaksu.

Instrukcije:

— KEY n,"tekst"
  — n — broj funkcijske tipke (1—16) — upis do 15 znakova
  — ovom naredbom funkcijskoj tipki dajemo značenje neke naredbe ili niza znakova
— DISPLAY
  — pokazuje što je pridruženo kojoj F-tipki
— AUTO p,k
  — p — početni broj linije; k — korak numeracije (1—255)
— RENUMBER p,k
  — p — početna nova numeracija; k — korak upozorenje: adrese u naredbama GOTO i GOSUB ne korigiraju se
— MERGE "ime",b
  — b — broj uređaja s kojeg učitavamo program (1 — kazeta, 8 — disk)
  — omogućuje dodatni upis programa bez brisanja prethodnoga programa iz memorije

— PAGE n
  — n — broj linija koje želimo odjednom izlistati (1—23)
  — listing se dijeli u stranice dužine n+1 linija, a nastavak listanja se postiže s RETURN
— OPTION n
  — n=10 uključuje naredbu, a n<>10 isključuje
  — prikazuje sve naredbe S. BASIC-a u listingu invertirano
— DELAY n
  — n — (1—255) — usporeno listanje programa
— FIND naredba, varijabla ili dio naredbe
  — dobivamo ispis linija koje sadržavaju dijelove iza naredbe FIND
— TRACE n
  — n=1Ø uključuje, n<>1Ø isključuje
  — na ekranu aktivira prozor u kojemu se prikazuje koje se naredbe upravo izvršavaju
— RETRACE
  — nakon brisanja ekrana možemo pozvati zadnji TRACE-prikaz
— DUMP
  — prikazuje sve varijable upotrijebljene u programu i njihove trenutačne vrijednosti
  — nedostaci:
    ne prikazuju se vrijednosti matrica
    negativne varijable se prikazuju kao pozitivne
    cijeli negativni brojevi prikazuju se tako da im se doda 2^{16}
    prazni nizovi su zauzeti sa 256 nedefiniranih znakova
— COLD
  — ovom se naredbom C-64 resetira
— OLD
  — regenerira program poništen s NEW ili COLD
— ON ERROR:GOTO k
  — k — broj linije na koju idemo u slučaju greške
— NO ERROR
  — vraća se kontrola grešaka BASIC-u
— DISAPA
  — radi u kombinaciji s naredbom SECURE Ø. Sa DISAPA označavamo retke koje želimo osigurati od listanja, tako da je upišemo ispred linije koju želimo zaštititi. Primjer: 2Ø DISAPA:LET B$="zaštita"
— SECURE Ø
  — daje se samo kao direktna naredba, te se nakon toga linije markirane sa DISAPA ne mogu listati
— IF ... THEN ... ELSE
  — IF uvjet, THEN linija za ispunjen uvjet, ELSE linija za neispunjen uvjet
  — naredba radi kao normalan IF ... THEN, uz granu ako uvjeti nisu ispunjeni

— RCOMP: linija za ispunjen uvjet. ELSE: za neispunjen uvjet
  — ako nakon IF . . . THEN . . . ELSE za isti uvjet ispitujemo. ne moramo ga ponavljati već uzmemo RCOMP
— REPEAT . . . UNTIL: uvjet
  — petlja se izvršava sve dok se ne ispuni uvjet
— LOOP . . . EXIT IF . . . END LOOP
  — format: LOOP, petlja: EXIT IF, uvjet: END LOOP
  — uvjet za napuštanje petlje nalazi se unutar petlje, a ne na kraju, kao u REPEAT . . . UNTIL
— PROC labela
  — definiramo labelu koju možemo pozvati sa CALL ili EXEC
— END PROC
  — definiramo kraj procedure inicirane s PROC; u biti nalik naredbi RETURN
— CALL labela
  — skok na proceduru — analogna naredbi GOTO
— EXEC labela
  — skok na proceduru definiranu s PROC i END PROC — analogna naredbi GOSUB
— CGOTO aritm. izraz
  — skok izračunan u aritm. izrazu
— LOCAL varijabla 1, var. 2, . . .
  — varijablama se pridaje lokalna vrijednost, a pri tome ostaje sačuvana prvobitna vrijednost
— GLOBAL
  — pozivanje prvobitne. globalne vrijednosti varijable
— MOD (dividend, divizor)
  — ispisuje ostatak cjelobrojnog dijeljenja
— DIV (dividend, divizor)
  — dobiva se cijeli dio kvocijenta
— FRAC (broj ili mat. izraz)
  — dobiva se decimalni dio broja ili izraza u zagradi
— EXOR (prvi broj, drugi broj)
  — korekcija prvog broja tako da mu se promijene bitovi koji su u oba broja isti
— % binarna vrijednost
  — pretvara zadani binarni broj u decimalni
— $ heksadec. vrijednost
  — pretvara hex-broj (4 mjesta) u decimalni broj
— INSERT (string koji umećemo, string u koji umećemo, pozicija na koju umećemo)
— INST (novi dio stringa, string koji mijenjamo, početna pozicija izmjene)
— PLACE (string koji se traži, string u kojem se traži)
  — ovom naredbom tražimo niz znakova unutar nekog drugog niza znakova
— DUP (string, faktor ponavljanja)
  — umnožavanje nekog niza znakova

— CENTRE string
  — centriramo izlaz stringa na ekran
— USE niz u obliku "## tekst #.##", niz s numeričkim vrijednostima
  — formiramo formatizirani izlaz numeričkih vrijednosti — npr. USE"##.##", "12.34534" ispisuje 12.34
— AT
  — format: PRINT varijabla 1 AT (stupac, redak) varijabla 2
  — ovom naredbom pozicioniramo izlaz varijable 2 na ekran, a varijablu 1 možemo i izostaviti
— LIN
  — PRINT LIN daje broj retka u kome se nalazi kursor
— PAUSE komentar, broj sekundi
  — pri nailasku na naredbu ispisuje se komentar, a izvođenje se zaustavlja za broj sekundi
— FLASH boja, brzina
  — postiže se treptanje ekrana u traženoj boji (Ø—15) traženom brzinom (Ø—255)
— OFF
  — isključuje naredbu FLASH
— BFLASH brzina treptanja, boja 1, boja 2
  — postiže se treperenje BORDER-a u bojama 1 i 2
— BFLASH Ø
  — isključuje naredbu BFLASH
— COLOUR boja okvira, boja pozadine
  — određuje se boja okvira i podloge
— BCKGNDS bpØ,bp1,bp2,bp3
  — ovom naredbom uključuju se proširene mogućnosti boje, pri čemu svaki znak može imati boju pozadine u jednoj od boja bpØ—bp3. Boja pozadine ovisi o ASCII-codu znaka. Za ØØØ—Ø63 bit će bpØ, za Ø64—127 bit će bp1, za 128—191 bit će bp2, i za 192—255 bp3. U ovom modu nema grafičkih znakova, inverznih i velikih slova.
— NMR
  — ovom naredbom isključujemo prošireni Color-mode. Osim toga ta naredba isključuje i GRAPHIC-mod i MEM-mod koje ćemo kasnije spomenuti.
— FCHR redak, stupac, visina, širina, znak
  — punjenje dijela ekrana istim znakom
— FCOL redak, stupac, visina, širina, boja
  — ovom naredbom svi znakovi na omeđenom području dobivaju traženu boju
— FILL redak, stupac, visina, širina, znak, boja
  — ispunjavanje dijela ekrana zadanim znakom u zadanoj boji
— MOVE redak, stupac, visina, širina, novi redak, novi stupac
  — ovom naredbom duplicira se dio ekrana na novo mjesto
— INV redak, stupac, visina, širina
  — invertiranje dijela ekrana (negativ-slika)

— LEFTW ili LEFTB
format: LEFTW redak, stupac, visina širina
— pomiče se dio ekrana ulijevo, a dio slike koji nestaje pojavljuje se na desnom rubu ekrana.
LEFTB ima isti format kao LEFTW, ali se slika ne pojavljuje na desnom rubu ekrana

— RIGHTW ili RIGHTB
— sve isto kao LEFT, ali je pomak udesno

— UPW ili UPB
— isti format kao LEFT, s time da je pomak prema gore

— DOWNW ili DOWNB
— isti format kao LEFT, s time da je pomak prema dolje
— za sve naredbe koje uključuju stupce i retke — broj stupca je Ø—39, a retka Ø—24

— FETCH "kontrolni znak", duljina, varijabla
— ovom naredbom možemo unositi podatke pod kontrolom. U varijablu se stavlja sadržaj neke duljine, ali tako da su dopušteni samo oni znakovi koji se nalaze u navodnicima. Npr. FETCH "Ø12",5,A prihvaća peteroznamenkaste brojeve sastavljene od cifara Ø, 1 i 2 i stavlja ih u varijablu A
— umjesto posebnog nabrajanja znakova mogu se upotrijebiti kontrolni znakovi:
CHR$(19) — prihvaća samo velika slova
CHR$(17) — prihvaća samo brojeve i posebne znakove
CHR$(29) — prihvaća samo velika slova sa SHIFT-om ili bez njega

— INKEY
— dobivamo odgovor koja je tipka F aktivirana. Ako nije nijedna odgovor je Ø.

— RESET broj DATA-retka
— bira se određeni DATA-redak iz kojega će se čitati podaci naredbom READ

— DISK "floppy-naredba"
— omogućuje kraći način saobraćanja s diskom. Npr. naredba DISK"N: ime diskete, oznaka diskete" zamjenjuje ovu seriju naredbi:
OPEN 15,8,15
PRINT#15, "N: ime diskete, oznaka diskete"
CLOSE 15
— isto se skraćuju i ostale naredbe za disk

— DIR format:
DIR"$"
DIR"$: ime datoteke ili dijela datoteke"
DIR"$??: ime datoteke ili dijela datoteke"
DIR"$: ✳=tip datoteke"
P — program
S — sekv. datoteka
R — relativna datoteka
U — USER-datoteka

— ovom se naredbom na ekranu ispisuje sadržaj diskete, a pri tom ne briše trenutačno prisutan program BASIC u memoriji
— SCRSV broj datoteke, broj uređaja, sekundarna adresa, "ime"
 — ovom naredbom sprema se ekran s tekstom na kazetu ili disk
— SCRLD broj datoteke, uređaja, sek. adresa, "ime"
 — pozivanje ekrana (LOAD) koji je spremljen naredbom SCRSV
— COPY
 — poziva se kopija ekrana na pisaču u visokoj rezoluciji. Kopija se dobiva na ovim pisačima:
  — CBM VC-1515
  — CBM VC-1525
  — SEIKOSHA GP 100 VC
  — EPSON RX 80 (sa DATA BECKER interfaceom)
  — EPSON FX 80 (sa DATA BECKER interfaceam)
  — CBM MPS-801
— HRDCPY
 — dobiva se kopija ekrana niske rezolucije na pisaču. Uz već nabrojene pisače može se upotrijebiti i CBM VC-1526
— HIRES boja točke (Ø—15), boja podloge (Ø—15)
 — ovom naredbom izvodimo inicijalizaciju fine grafike na ekranu. Za isključenje grafike u toku rada programa služi naredba NRM ili CSET Ø/1
— MULTI boja 1 (Ø—15), boja 2 (Ø—15), boja 3 (Ø—15)
 — naredba uključuje MC-mod i istodobno definira 3 boje za upotrebu na ekranu (napomena: naredbi MULTI mora prethoditi naredba HIRES)
— NRM
 — isključujemo grafički mod i uključujemo tekst mod (velika slova/grafički znaci CBM)
— CSET 2
 — ova naredba uključuje grafiku na ekranu bez brisanja grafičke stranice iz RAM-a
— LOW COL
 (za HGR mod): LOW COL boja (Ø—15)
 (za MC mod): LOW COL boja 1 (Ø—15), boja 2 (Ø—15), boja 3 (Ø—15)
— HI COL
 — isključuje naredbu LOW COL, a crtanje se nastavlja u zadnjoj (HGR), odnosno zadnje 3 boje (MC-mod)
— PLOT x,y, način crtanja
 — crta se točka na mjesto određeno koordinatama x,y. Ishodište je u gornjem lijevom kutu ekrana, a vrijednosti za x,y koordinate su:
 u HGR-modu — x(Ø—319) y(Ø—199)
 u MC-modu — x(Ø—159) y(Ø—199)

— načini crtanja su za HGR-mod:
  Ø — brisanje (crtanje u boji podloge)
  1 — obično crtanje
  2 — invertirano (negativ) crtanje
— za MC mod:
  Ø — brisanje crteža
  1 — boja 1
  2 — boja 2
  3 — boja 3
  4 — invertirano crtanje

— TEST (x,y)
  — ispituje se da li je na mjestu x,y nacrtana točka. Ako točka postoji odgovor je 1, a ako ne postoji odgovor je Ø.

— LINE x1,y1,x2,y2, način crtanja
  — ovom naredbom crta se linija između dvije zadane točke x1,y1 i x2,y2

— REC x,y, širina, visina, način crtanja
  — crta se pravokutnik zadane širine i visine, kome je gornji lijevi kut u točkama x,y

— BLOCK x1,y1,x2,y2, način crtanja
  — crta se pravokutnik s gornjim lijevim kutom u x1,y1, donjim desnim kutom u x2,y2 i zatim se ispuni ovisno o načinu crtanja

— CIRCLE sx,sy,rx,ry, način crtanja
  — crta se elipsa sa središtem u sx,sy i s radijusima rx,ry. Kružnica se dobiva kada je rx=ry (HGR) ili rx/2=ry (MC)

— ARC sx,sy,pk,kk,k,rx,ry, način crtanja
  — crta se dio luka elipse sa središtem u sx,sy, radijusima rx,ry. Luk se počinje crtati na mjestu zadanom početnim kutom (pk), a završava na mjestu zadanom krajnjim kutom (kk). Broj točaka luka određuje se parametrom k, koji je najčešće k=12
  — kutovi su zadani ovako:

```
          Ø
     315     45
 27Ø             9Ø
     225    135
         18Ø
```

— ANGL sx,sy,p,rx, način crtanja
  — ovom naredbom crta se radijus zadane elipse na položaju p (kut Ø—36Ø), a ostali parametri su isti kao u prethodnim naredbama

— PAINT x,y, način crtanja
  — popunjavanje površine unutar koje se nalazi zadana točka, ovisno o načinu crtanja

— DRAW str,x,y, način crtanja
  — ovom naredbom crta se slika koja je definirana u string-varijabli str, ili nizom brojki u navodnicima. Slika se počinje crtati od koordinate x,y, a brojke u navodnicima imaju ovo značenje:

BROJ	SMJER POMAKA	CRTANJE TOČKE
Ø	desno	ne
1	gore	"
2	dolje	"
3	lijevo	"
4	dolje	"

5	desno	da
6	gore	,,
7	dolje	,,
8	lijevo	,,
9	prekid crtanja	

— ROT kut rotacije, faktor povećanja

— ovom naredbom povećavaju se i rotiraju figure formirane naredbom DRAW. Kut rotacije može imati vrijednosti od Ø do 7, a faktor povećanja Ø—255. Faktor 1 ne povećava figuru, a faktor Ø povećava 256 puta.

BROJ	KUT ROTACIJE
Ø	Ø
1	45
2	9Ø
3	135
4	18Ø
5	225
6	27Ø
7	315

— rotacija se izvodi oko početne točke u desno

— CHAR x,y, kod znaka, način crtanja, visina znaka

— ovom naredbom crta se samo jedan znak na ekran, određen njegovim kodom (Ø—255). Za dobivanje inverznog znaka, kodu se dodaje 128. Znak se može povećati po visini (Ø—255), ali ne i po širini. Gornji lijevi kut znaka je definiran varijablama x,y.

— TEXT x,y,str, način crtanja, visina znaka, razmak znakova

— ovom naredbom ispisuje se tekst koji se nalazi u str(ing)-varijabli ili između navodnika. Sa x,y je označen lijevi gornji kut prvog znaka u nizu, dok se visina definira isto kao u naredbi CHAR. Razmak označava razmak između lijevih rubova znakova, kojima je širina 7 točkica, pa će se za brojeve manje od 7 znakovi preklapati. Naredbom TEXT se mogu pisati velika i mala slova, ovisno o tome stisnemo li CTRL A ili CTRL B

— MEM

— ova naredba izvodi pet radnji:

— kopira 4KB ROM-a s definiranim znakovima ($DØØØ—$DFFF) u RAM na adresno područje $EØØØ—$EFFF

— mijenja se pokazivač koji govori kompjuteru gdje da traži znakove

— videoRAM seli u drugo područje s početkom na adresi $CCØØ

— briše se ekran

— područje za spriteove seli se na adresu $CØØØ

— ovim operacijama omogućuje se mijenjanje i definiranje novih znakova

— DESIGN 2, adresa
  — adresa=$EØØØ +8xke; ke=kod ekrana
  — ovom naredbom označavamo znak koji želimo mijenjati
— ∂ točkice i slova B (slovom ∂ u ovom priručniku označen je kompjuterski znak koji se nalazi u drugom retku tastature, četvrti s desna)
  — ova naredba slijedi iza DESIGN 2, a služi za definiranje jednog reda od 8 točkica matrice pomoću koje formiramo novi znak. To znači da nam za cijeli znak treba 8 uzastopnih ∂ naredbi
— CSET Ø ili CSET 1
  — naredbom CSET Ø biramo grupu znakova A (velika slova i grafički znaci), a sa CSET 1 grupu B (mala i velika slova)

Instrukcije koje služe pri definiranju spriteova:

— DESIGN Ø/1
  format:
    DESIGN Ø,ba+bx64
    DESIGN 1,ba+bx64
    Ø-sprite u HGR-modu
    1-sprite u MC-modu
    ba=Ø-tekst-mod
    ba=$CØØØ — novi znakovi
    b=Ø—255 — broj bloka memorije u koji se sprema definicija spritea
  — ovom naredbom definiramo prije formiranja spritea, kakav sprite želimo (HGR ili MC) i gdje ga spremamo
— ∂ niz točaka i slova B, C ili D
  — značenje
    . prozirno
    A točka boje 1
    B točka boje 2
    C točka boje 3
  — nakon naredbe DESIGN slijedi 21 naredba ∂, pomoću kojih definiramo sprite. U HGR-modu služe samo točka i slovo B, a u MC-modu mogu se uzeti 3 boje. U HGR--modu, naredbu ∂ slijede 24 znaka, a u MC-modu 12 znakova
— MOB SET, n,b, boja, prioritet, rezolucija
    n — broj spritea (Ø—7)
    b — broj bloka u kojem je sprite
    boja — (Ø—15) — HGR-B, MC-C
    prioritet — Ø — ispred pozadine
                1 — iza pozadine
    rezolucija — Ø — sprite u HGR-modu
                 1 — sprite u MC-modu
  — ovom naredbom završava se definicija ako je u HGR-modu
— CMOB f1, f3
    f1 — boja 1 (Ø—15), slovo B za MC sprite
    f2 — boja 3 (Ø—15), slovo D za MC sprite
  — ovom naredbom završava se kreiranje spritea u MC-modu

— MMOB n,x1,y1,x2,y2, veličina, brzina
  n (Ø—7) — broj spritea
  x1 (Ø—511) — x koordinata starta
  y1 (Ø—255) — y koordinata starta
  x2 (Ø—511) — x koordinata cilja
  y2 (Ø—255) — y koordinata cilja
  — te varijable odnose se na lijevi donji kut spritea
  veličina ima ove vrijednosti:
  Ø — standardna 24/21 točkica
  1 — dvostruko šira 48/21 točkica
  2 — dvostruko viša 24/42 točkica
  3 — dvostruko veća 48/42 točkica
  brzina (Ø—255) spritea između starta i cilja
  — ovom naredbom pozivamo određeni sprite, dajemo mu određenu veličinu i brzinu, te ga pomičemo od jedne do druge točke

— MOB OFF n
  — brisanje spritea broj n s ekrana, ali sprite ostaje definiran i može se pozvati naredbom MMOB

— RLOCMOB n,x2,y2, veličina brzina
  — pomicanje spritea na ekranu u točku x2, y2

— DETECT s
  s — Ø sudar sprite-sprite
  s — 1 sudar sprite-lik pozadine
  — ispitivanje sadržaja registra za detekciju sudara — ovisno o sadržaju ispisuje se da li se sprite sudara s drugim spriteom ili likom pozadine

— CHECK (n1,n2)
  CHECK (n)
  — između naredbe i zagrade ne smije biti prazno mjesto
  — ispituje se da li je sudar bio s drugim spriteom ili pozadinom — CHECK (n). Ovisno o naredbi, prethodi im DETECT Ø ili DETECT 1. Bez sudara CHECK ima vrijednost 1, a sa sudarom poprima vrijednost Ø.

Instrukcije kojima se manipulira zvukom:

— VOL jačina glasa (Ø—15)
  — mijenjanje jačine zvuka za sva tri kanala; nije moguće pojedinačno određivanje

— WAVE glas (1—3), oblik vala
  — definiranje oblika vala za pojedini kanal. Oblik se definira kao broj od 8 znamenki, koje imaju vrijednost 1 ili Ø. Riječ je o 8 bitova, a značenje im je:

BIT	FUNKCIJA
Ø	isključivanje (Ø), uključivanje (1) kanala
1	sinhronizacija kanala — sinhroniziraju se 1. i 3. kanal, 2. i 1. ili 3. i 2. kanal
2	prstenasta (kružna) modulacija
3	bit za testiranje

4 izbor trokutnog valnog oblika
5 izbor pilastog valnog oblika
6 izbor pravokutnog valnog oblika
7 izbor šuma

— ENVELOPE kanal (1—3), a,d,s,r
- — parametri ovojnice (envelope)
  a (Ø—15) vrijeme porasta
  d (Ø—15) vrijeme pada
  s (Ø—15) vrijeme održavanja
  r (Ø—15) vrijeme opuštanja
- — definiranje tona s obzirom na jačinu

— MUSIC ritam, $ varijabla ili konstanta s notama
- — ritam (Ø—255) daje brzinu izvođenja nota
- — ovom naredbom definira se niz nota za neku kompoziciju. Ritam se definira po formuli: ritam/12. To znači da će za ritam 6 nota trajati 1/2 sekunde itd. Note se definiraju tako da se tastatura prvo prebaci u mod za velika i mala slova pritiskom na SHIFT i COMMODORE, a ako to ne uspije, tada PRINT CHR$(9) i RETURN. Nakon toga treba definirati kanal, tako da odmah iza navodnika stisnemo SHIFT i CLR/HOME i iza toga broj kanala (1—3). Note su c,d,e,f,g,a,b, a ako želimo povišeni ton stisne se SHIFT i nota. Oktava u kojoj se nalazi ton definira se od Ø do 7. Najviši ton je A7. Zatim treba definirati trajanje note pomoću funkcijskih tipki:

TIPKA	TRAJANJE	OZNAKA NA EKRANU (INVERZNO)
F1	1/16	E
F3	1/8	F
F5	1/4	G
F7	1/2	H
F2	1/1	I
F4	2/1	J
F6	4/1	K
F8	8/1	L

- — ako se u toku muzike želi prekid, upisuje se SHIFT i CLR/HOME i iza toga g. Veća muzička djela mogu se raditi tako da se spajaju nizovi:
  npr. MUSIC 10,A$+B$+C$, koji naravno prethodno moraju biti definirani

— PLAY način (1—2)
- — ovom naredbom muzika se izvodi, i to uz način 1 tako da muzika svira, a program ne traje, ili način 2, da program traje uz muziku

— PENX (funkcija)
- — dobiva se x koordinata točke označene pomoću lightpena

— PENY
- — dobiva se y koordinata točke označene pomoću lightpena

— POT (broj uređaja — Ø ili 1)
- — dobiva se trenutačna vrijednost funkcije POT

— JOY
  — dobiva se položaj palice za igru (joysticka) prema shemi:
    8 1 2
    7 Ø 3
    6 5 4 128 — fire (paljba)
  — priključenje na port 2

DOJAVA GREŠAKA

U toku pisanja programa ili unosa podataka mogu se napraviti neke greške, koje računalo locira i na ekranu daje upozorenje. Ovdje je kompletna lista poruka za greške i njihovo objašnjenje.

BAD DATA
  — program je očekivao numeričke podatke, a dobio je string (tekst) podatke

BAD SUBSCRIPT
  — program je pokušao složiti element polja čiji je broj izvan granica propisanih za upotrebu u naredbi DIM

BREAK
  — program zaustavljen pritiskom na tipku STOP

CAN'T CONTINUE
  — program se ne može nastaviti, jer ili nije uopće startan, ili je greška u programu, ili je editiran

DEVICE NOT PRESENT
  — traženi uređaj nije na raspolaganju za OPEN, CLOSE, CMD, PRINT#, INPUT# ili GET#

DIVISION BY ZERO
  — dijeljenje s nulom je matematički nonsens i nije dopušteno

EXTRA IGNORED
  — previše podataka je uneseno kao odgovor na zahtjev INPUT

FILE NOT FOUND
  — ako tražite na vrpci pronađen je marker za kraj vrpce, a na disku ne postoji datoteka s tim imenom

FILE NOT OPEN
  — niz specificiran naredbama CLOSE, CMD, PRINT#, INPUT#, GET#, mora prvo biti otvoren naredbom OPEN

FILE OPEN
  — pokušaj da se otvori niz (file) brojem koji već ima otvoreni niz

FORMULA TOO COMPLEX
  — izraz bi trebao biti razdijeljen u najmanje dva dijela da bi sistem mogao s njime raditi

ILLEGAL DIRECT
  — naredba INPUT može se primijeniti samo unutar programa, a ne kao direktna naredba

ILLEGAL QUANTITI
— broj upotrijebljen kao argument funkcije ili naredbe je izvan dopuštenih granica

LOAD
— problem s programom na vrpci

NEXT WITHOUT FOR
— uzrok je nepravilno postavljanje petlji ili postojanje naredbe NEXT bez odgovarajuće naredbe FOR

NOT INPUT FILE
— pokušaj da se radi s naredbama INPUT ili GET s nizom koji je predviđen samo kao izlazni

NOT OUTPUT FILE
— pokušaj da se naredbom PRINT upišu podaci u niz predviđen samo za čitanje

OUT OF DATA
— pri izvršavanju naredbe READ nestalo je podataka u DATA-liniji

OUT OF MEMORY
— nema više raspoloživog RAM-a za program ili varijable. Uzrok također može biti previše petlji FOR ... NEXT ili previše naredbi GOSUB

OVERFLOW
— rezultat je veći od najvećeg dopuštenog broja: 1.70141884E+38

REDIM'D ARRAY
— polje se smije dimenzionirati samo jednom

REDO FROM START
— u INPUT su upisani karakteri umjesto očekivanih brojeva. Trebate samo ponovno upisati ispravne brojeve i program nastavlja izvršenje

RETURN WITHOUT GOSUB
— pronađena naredba RETURN, a da nije nigdje upotrijebljena naredba GOSUB

STRING TOO LONG
— string može sadržavati do 255 karaktera

?SYNTAX ERROR
— naredba neprepoznatljiva COMMODORE-ovom operacionom sistemu. Uzrok nedostatak ili višak parametara, krivi upis itd.

TYPE MISMATCH
— greška koja nastaje kad se upotrijebi broj umjesto stringa i obratno

UNDEF'D FUNCTION
— korisnička funkcija je pozvana, ali nije nigdje definirana naredbom DEFFN

UNDEF'D STATEMENT
— pokušaj da se pozove linija koja ne postoji naredbama GOTO, GOSUB ili RUN

VERIFY
— program na vrpci ili disku ne odgovara programu u memoriji

GRAFIČKE MOGUĆNOSTI C 64

GRAFIKA I BOJA NA C64

Na Commodoreu 64 na raspolaganju stoji grafika niske i visoke razlučivosti, sprite-grafika te upotreba 8 osnovnih boja iz kojih se pod određenim uvjetima može dobiti 16 nijansi. Upotreba grafike je ponešto otežana zbog upotrebe instrukcija POKE i PEEK.

Na ovom mjestu ćemo se pozabaviti grafikom niske razlučivosti, a treba reći da se ona dobiva upotrebom standardnih znakova. Pri radu s »niskom grafikom« ekran je podijeljen na 1000 lokacija znakova, odnosno na 25 linija od po 40 znakova (karaktera). Svi ti znakovi su pohranjeni u memoriji ROM — u dvije odvojene grupe (seta). Istodobno se ne može raditi s oba seta, a »preklapaju« se istodobnim pritiskanjem tipaka SHIFT i COMMODORE.

Sadržaj prvog seta su velika slova, znamenke, neki posebni znakovi koji se upotrebljavaju u BASIC-u te neki grafički znakovi. Taj set je dostupan za upotrebu odmah po uključenju računala.

Drugi set sadržava samo mala i ostatak velikih slova.

Dva su načina izrade grafike niske razlučivosti — upotrebom instrukcija PRINT i CHR$. PRINT otvara mogućnost dobivanja grafike i jednostavne animacije. Za animaciju treba programski omogućiti kontrolu gdje će biti prisutan slijedeći znak na ekranu.

Upotreba grafike niske razlučivosti nije problem. ali nije ni osobito zanimljiva kao grafika visoke razlučivosti koju ćemo objasniti malo poslije.

Za one slabije upućene treba reći da se u »niskoj grafici« vrlo teško ili gotovo nikako ne mogu izvesti precizni crteži. a »visokom grafikom« mogu se postizati precizne odnosno vrlo fine konture likova.

CHR$ KODOVI I GENERIRANJE BOJE

CHR$ (X) je string-funkcija raznolike primjene. Pomoću nje se na ekranu mogu dobiti znakovi čija je vrijednost po ASCII-kodu vrijednost X u zagradi. ASCII-kod (American Standard Code for Information Interchange) usvojila je većina proizvođača računala. Standardni ASCII obuhvaća 128 znakova (0—127). Prva 32 znaka su posebni kontrolni znakovi, uglavnom za kontrolu terminala ili printera. Slijedećih 32 znaka su znamenke od 0 do 9, te posebni znakovi iz BASIC-a kao npr.: (.), +, $ itd.

Na C64 se nešto odstupa od standardnog ASCII-koda. tj. grupe 2 i 3 od po 32 znaka obuhvaćaju znamenke, slova i posebne znakove, a četvrta grupa slova i neke posebne kontrolne kodove.

Osnovni oblik instrukcije je CHR$ (X) gdje je X vrijednost znaka, boje ili nečeg drugog. Tom funkcijom može se pomicati i kursor. Tako se brisanje ekrana i postavljanje kursora na početak postiže s CHR$ (147), micanje lijevo-desno sa CHR$ (157) odnosno CHR$ (29), a pomaci gore-dolje sa CHR$ (145) i CHR$ (17).

Ako za CHR$ upotrijebimo vrijednost X = 113 dobit ćemo znak — malu lopticu (kursor) koja služi u ovom programu:

```
  5 A=1Ø : B=1Ø
 15 E=1 : F=1
 25 PRINT CHR$ (147)
 35 FOR D=1 TO A : PRINT CHR$ (29); :
    NEXT D
 45 FOR D=1 TO B : PRINT CHR$ (17); :
    NEXT D
 55 PRINT CHR$ (157);" ";CHR$ (157);
 65 IF E=1 THEN PRINT CHR$ (29); :
    GOTO 85
 75 PRINT CHR$ (157);
 85 IF F=1 THEN PRINT CHR$ (17); :
    GOTO 1Ø5
 95 PRINT CHR$ (145);
1Ø5 PRINT CHR$ (113);
115 A=A+E : B=B+F
125 IF A<2 OR A>39 THEN E=—E
135 IF B<2 OR B>24 THEN F=—F
145 GOTO 55
```

Kada utipkamo taj program i pokrenemo ga s RUN, dobit ćemo dijagonalno kretanje loptice (kursora) svakuda po ekranu.

Objašnjenje programa:

A i B su dvije varijable koje daju smjer tekuće pozicije loptice, a linija 5 ih zaustavlja na 10,10. E i F su varijable koje daju smjer kada loptica putuje. Ako je E pozitivno loptica će se kretati desno, a obratno lijevo. Ako je F pozitivno loptica se kreće prema dolje, a ako je F negativno ide prema gore.

Linija 15 starta lopticu, a linija 25 briše ekran i postavlja kursor. Linije 35 i 45 pomiču kursor uz instrukciju CHR$ (X) za pokretanje. Linija 55 »briše« lopticu za ponovni prikaz na nekom drugom mjestu, odnosno dok se proračuna njezina nova pozicija. Linijama 65 i 75 kursor se pokreće lijevo ili desno ovisno o varijabli E, a u linijama 85 i 95 gore ili dolje ovisno o varijabli F. Linija 105 daje loptici novu poziciju, a linija 115 daje nove vrijednosti varijablama A i B. Linije 125 i 135 provjeravaju vrijednosti varijabli A i B. Naposljetku linija 145 vraća program petljom na liniju 55.

Taj je program osnovica za mnoge igre s loptom za koje isto tako služi funkcija CHR$ (X).

Boja

C64 može prikazati 16 nijansi boja, ali samo 8 boja uz funkciju CHR$. Za ostale nijanse mora se posegnuti na određene memorijske lokacije naredbom POKE, odnosno unositi vrijednosti na te lokacije.

Boje i njihovi CHR$-kodovi izgledaju ovako:

BOJA	KOD
bijela	5
crvena	28
zelena	30
plava	31
crna	144
purpurna	156
žuta	158
cijan	159

Za demonstraciju kontrolnih kodova boje može poslužiti primjer:

```
5 REM primjer za demonstraciju boje
25 RESTORE : POKE 53281,11
35 FOR A=1 TO 8
37 READ B
42 PRINT  CHR$(B);  "DEMONSTRACIJA
   BOJE"
45 NEXT A
55 DATA 5,158,159.3Ø,156,28,31,144
```

Ako ga startate vidjet ćete drugačiju pozadinu, a to je proizvedeno s POKE u liniji 25. Što se tiče nijansi boja i POKE-iranja njihovih vrijednosti, treba reći da one zauzimaju memorijske lokacije od 55296 do 56295 (decimalno). To je prostor od 1000 byteova jer je upravo toliko moguće prikazati znakova na ekranu (25×40=1000). Dakle, na ekranu se svaki od 1000 znakova može obojiti u jednu od 16 mogućih nijansi.

Kolor-mapa 1000 bytea izgleda: Byte 55296 je kolor byte za gornju lijevu lokaciju na ekra-

nu. U tom prvom redu najdesniji byte je 55335. Byte 55336 je prvi byte na lijevoj strani u drugoj liniji. Posljednji kolor-byte je u donjem desnom kutu ekrana — byte 56295. Dakle, svaka pozicija na ekranu može se prikazati sa X i Y.

Koordinata X može imati vrijednosti od 0 do 39. a Y od 0 do 24. Poznavajući vrijednosti X i Y moguće je izvesti proračun kolor-mape byteova kao:

MB = 55296 + 40*Y + X

Mapa byteova je proračunana kao adresa prvog bytea 55296 plus Y kompletnih linija od po 40 znakova prema dnu ekrana, plus X znakova unutar željene linije.

Vrijednosti (nijanse) koje se mogu POKE-om unijeti u kolor-mapu iznose prema tablici:

0 = crno	8 = narančasto
1 = bijelo	9 = smeđe
2 = crveno	10 = svijetlocrveno
3 = cijan	11 = siva 1
4 = purpurno	12 = siva 2
5 = zeleno	13 = svijetlozelena
6 = plavo	14 = svijetloplava
7 = žuto	15 = siva 3

Saznali smo kako se može promijeniti boja svakom znaku na ekranu, ali što je s bojom pozadine?

Nijansa pozadine određena je sadržajem memorijske lokacije 53281. U tu lokaciju može se unijeti 16 gornjih vrijednosti. Boja okvira (border) određuje se unošenjem vrijednosti na lokaciju 53280. Tako, ako npr. želimo zeleni okvir utipkamo:

POKE 53280,5

Potpuno savladavanje instrukcije POKE postiže se samo pisanjem vlastitih rutina.

UPOTREBA PEEK I POKE

Govoreći o boji pokazali smo primjer instrukcije POKE za određivanje nijansi. Pogledajmo što su zapravo te instrukcije.

Za mnoge početnike ove instrukcije znače kobnu prepreku, a oni vještiji ih nastoje upotrebljavati što manje, odnosno samo za posebne namjene. Bez Simon's BASIC-a uspješna upotreba grafičkih mogućnosti Commodorea 64 je nezamisliva ako se ne upotrebljavaju instrukcije PEEK i POKE.

Što su zapravo te naredbe?

Instrukcija POKE nije ništa drugo do upisivanje neke vrijednosti na određenu memorijsku lokaciju, a instrukcija PEEK služi za iščitavanje vrijednosti s memorijskih lokacija. Ispravan upis i oblikovanje instrukcija izgleda npr.

POKE 55555,33 odnosno PEEK 55555

Memorijske lokacije su cijeli brojevi od 0 do 65535 (zbog 64K memorije), a moguće vrijednosti parametara su od 0 do 255. Zbog kon-

strukcije i namjene u ROM-memoriju ne mogu se unositi nikakve nove vrijednosti (npr. u adresama od 40960 do 49151). U neke lokacije nije preporučljivo unositi vrijednosti POKE jer tada računalo »zablokira«. To ga ne može pokvariti, ali uzrokuje smetnje i greške u programu.

Naredbom POKE znakovi se mogu smještati direktno na ekran, a isto tako se može definirati i boja znaka. Također nalaze svoju primjenu u iskorištenju zvukovnih mogućnosti na C64.

Definiranje boje smješteno je na memorijskim lokacijama 55296 do 56295. Na primjer, želimo li na mjestu X,Y smjestiti slovo A u crvenoj boji utipkamo:

```
POKE Y*4Ø+X+1Ø24,1 :
POKE Y*4Ø+X+55296,2
```

Broj 1 u prvom dijelu dolazi od činjenice da je kôd slova A broj jedan, a u drugom dijelu kôd crvene boje je broj 2. Isto tako, ako iz nekog razloga želimo obojiti pozadinu ekrana u crno, unosimo u računalo:

```
POKE 5328Ø,Ø : POKE 53281,Ø.
```

Instrukcije POKE i PEEK mogu imati najraznovrsniju primjenu. Na primjer »prekapčanje« računala na mala slova sa:

```
POKE 53272,23
```

ili vraćanje na velika slova:

```
POKE 53272,21
```

Onima koji su još skeptični prema ovim instrukcijama treba reći da su u POKE-ovima, točnije njihovim vrijednostima, skriveni »životi« koje imaju u pojedinim igrama.

Promjenom tih varijabli, igre se mogu prilagoditi refleksima igrača. Dakle, nema ni igara bez »pikanja« (PEEK i POKE) po računalu.

SPRITE GRAFIKA

Što je zapravo sprite (sprajt)?

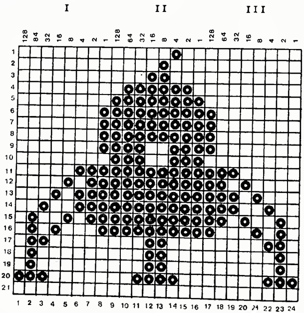
Najjednostavniji odgovor bio bi: to je mali dio slike na ekranu, sastavljen od točkica visoke razlučivosti. Može se pokretati kao cjelina preko ekrana pri čemu se mora definirati njegov novi položaj. Pri tome ne treba brinuti o pojedinim točkicama od kojih je sprite sastavljen. S obzirom na tu činjenicu imamo velike mogućnosti kreiranja slika jednostavnim rutinama u BASIC-u. Pretapanjem spriteova mogu se postizati zanimljivi trodimenzionalni efekti.

Za upotrebu spriteova postoji nekoliko koraka: oblikovanje spritea, pretvorba spritea u brojeve, spremanje dobivenih brojeva u memorijske lokacije, naznaka računalu gdje će se ti brojevi spremiti, određivanje boje spritea, naznaka računalu gdje će se sprite prikazati na ekranu te uključivanje spritea.

Na Commodoreu 64 sprite je oblikovan kao kvadratić od 24 × 21 točkice visoke razlučivosti. Spriteovima upravlja poseban čip u računalu

— VIC (Video Interface Controller). Čip VIC može kontrolirati do 8 različitih spriteova. Za grupiranje podataka o spriteovima predvidene su memorijske adrese 2040 do 2047, tako da je svaki registar na pojedinoj adresi-lokaciji podatak za jedan sprite. Za svaki sprite je podatak dugačak 63 bytea, a ti podaci se smještaju u prvih 16K memorije. Takoder se može upotrije-

PLANIRANJE SPRITE-a



Primjer oblikovanja SPRITE-a. Iz mreže točaka treba izvaditi vrijednosti koje se potom unose u DATA programske linije prema primjeru

DATA

I	II	III
0	4	0
0	8	0
0	24	0
0	126	0
0	255	0
1	255	128
1	255	128
1	255	128
1	231	128
0	231	0
7	255	224
11	255	208
23	255	232
39	265	228
75	255	210
81	255	138
96	24	6
64	24	2
84	24	2
224	60	7
0	0	0

biti i kazetofonski ulazno/izlazni spremnik (lokacije 828 do 1023), te područje za BASIC (2048 do 40959). To je obično zauzeto BASIC-om, ali se rutinom može osigurati malo prostora za sprite. Rutina će osloboditi 512 byteova:

```
NEW
POKE 642,1Ø : POKE 44,1Ø : POKE 46,1Ø :
POKE 48,1Ø : POKE 5Ø,1Ø : POKE 256Ø,Ø
NEW
```

Spriteovi — točnije njih osam, mogu se kontrolirati registrom — Sprite Enable register, smještenim na memorijskoj lokaciji 53269.

Kao primjer definiranja može poslužiti kratki priloženi program, koji će od lokacije 832 spremati podatke za konačno oblikovanje spritea.

```
5 REM primjer za oblikovanje spritea
1Ø REM definiranje FOR/NEXT petlje
15 FOR A = Ø TO 63
2Ø REM vrijednost za POKE
25 READ B
3Ø REM POKE vrijednosti na slijedeću lokaciju
35 POKE 832 + A,B
4Ø REM ponavljanje za slijedeću vrijednost
45 NEXT A
5Ø END
55 REM podaci za sprite
6Ø DATA Ø,4,Ø,Ø,8,Ø,Ø,24
65 DATA Ø,Ø,126,Ø,Ø,255,Ø,1
7Ø DATA 255,128,1,255,128,1,255,128
75 DATA 1,231,128,Ø,231,Ø,7,255
8Ø DATA 224,11,255,2Ø8,23,255,232,39
85 DATA 255,228,75,255,21Ø,81,255,138
9Ø DATA 96,24,6,64,24,2,64,24
95 DATA 2,224,6Ø,7,Ø,Ø,Ø,Ø
```

Bojenje spriteova

C64 raspolaže sa 16 boja-nijansi za spriteove. Njihove vrijednosti za POKE već su navedene u cjelini o CHR$-funkciji i boji. Ovisno o broju boja spritea postoje dva moda rada. Standardni kolor-mod, u kojemu sprite može biti u samo jednoj boji na transparentnoj pozadini i tzv. multi-kolor mod kojim se sprite može obojiti u tri boje s transparentnom pozadinom.

Pod pojmom transparentne pozadine podrazumijevaju se sve točke koje ne »sjaje« na spriteu već se nalaze uokolo. Boja spriteova se u standardnom kolor-modu odabire vrijednošću uz instrukciju POKE te se tako unose u Sprite Colour registre smještene na lokacijama 53287 do 53294. Raspoređeni su po pojedinom spriteu — registar 53287 za sprite Ø, 53288 za sprite 1, 53289 za sprite 2, itd.

Efektivno u svakom tom registru sudjeluju samo 4 bita jer se njima može postići 16 potrebnih kombinacija (2^4), odnosno 16 nijansi boja. Kada se znaju lokacije registara i željene vrijednosti boja možemo npr. sprite broj 1 obojiti u smeđe:

```
POKE 53288,9
```

a na primjer sprite br. 3 u crvenu boju:

POKE 53290,2

U multikolor-modu vodoravne točke očitavaju se u parovima, a svaki par točaka može biti u jednoj od tri boje ili »transparentu«. Tablica pokazuje kako su parovi točaka kodirani u boje:

BOJA UZETA SA:	PAR TOČAKA
— transparenta	ØØ
— sprite multikolor-registra Ø	Ø1
— normalnog sprite kolor-registra	1Ø
— sprite multikolor-registra 1	11

Dva multikolor-registra podižu vrijednost dviju posebnih boja. Te su boje uvijek za sve spriteove u multikolor-modu. Multikolor-registri smješteni su s obzirom na lokacije: multikolor Ø na adresi 53285, a multikolor sprite 1 na adresi 53286. Multikolor-mod odabire se postavljanjem odgovarajućeg bita u multikolor-registar. Taj je smješten na adresi 53276. Bit Ø kontrolira sprite Ø, bit 1 kontrolira sprite 1 itd.

Kao primjer može se uzeti prikaz spritea 2, kombinacijom purpurne, zelene i plave boje:

```
5 REM postavljanje multikolor-moda za
  sprite 2
1Ø POKE 53276, PEEK (53276) OR 4
15 REM postavljanje sprite multikolor-regi-
  stara
2Ø REM za purpurnu boju — 4, za zelenu
  boju — 5
25 POKE 53285,4 : POKE 53286,5
3Ø REM postaviti sprite kolor-registar za pla-
  vu boju
35 POKE 53289,6
```

Pokretni spriteovi

Svoju pravu vrijednost spriteovi nalaze u animaciji. Pokreću se promjenom lokacije para registara nazvanih Sprite Position Register, za svaki pojedini sprite. Ti registri zauzimaju mjesto od gornjeg lijevog kuta ekrana (pri ispisu), u visokoj razlučivosti 320 × 200 točaka. Čip VIC neprestano »iscrtava« prikaz na ekranu te tako pronalazi sprite. Ako se s POKE unose nove vrijednosti u par registara, čip VIC će izazvati pomak spritea sa stare na novu poziciju. Tablica prikazuje lokacije Sprite Position registara:

X REGISTAR	BROJ SPRITEA	Y REGISTAR
53248	0	53249
53250	1	53251
53252	2	53253
53254	3	53255
53256	4	53257
53258	5	53259
53260	6	53261
53262	7	53263

Za visoku razlučivost u X-smjeru idu vrijednosti od Ø do 319, a u Y smjeru od Ø do 199. Što se spriteova tiče postoji mala razlika, naime zbog njihova pokretanja »uključuje« se i »isključuje«

se ekran bez potrebe za negativnim koordinatama pozicije.

To znači da je ekran podijeljen od 24 do 344 u X-smjeru i od 50 do 250 u Y-smjeru.

Problem je kada se za X uzmu veće vrijednosti, zbog toga što X Sprite Position registri mogu podnijeti samo vrijednosti od Ø do 255, a to nije dovoljno za popunjavanje adresa ekrana od 320 točaka. Tada treba uzeti poseban registar koji će omogućiti poseban bit, odnosno 9-bitnu širinu (9-bitni registar ima raspon vrijednosti od Ø do 511). Novi registar naziva se MSBX (Most Significant Bit X Register), a smješten je na lokaciji 53264. Uz njegovu pomoć, za X-koordinatu mogu se uzeti vrijednosti manje ili veće od 255. Ako je vrijednost manja od 255 mora se resetirati odgovarajući poseban bit i unijeti X-koordinata u pripadni registar za poziciju spritea.

Ako je vrijednost veća od 255 treba postaviti poseban (extra) bit, oduzeti 256 od koordinate, te pohraniti ostatak u pripadni X-registar za poziciju spritea.

Na slijedećem primjeru programa može se vidjeti kako se postavlja registar-pozicija spritea, poštujući veličinu X-koordinate. Broj spriteova mora biti prije poziva rutine smješten u varijablu A, a koordinate X i Y smještene kao vrijednosti varijabli X i Y.

```
5 REM rutina za postavljanje registra pozicije spritea
1Ø REM pohranjivanje Y-koordinate
15 POKE 53294 + A*2,Y
2Ø REM provjera veličine X-koordinate
25 IF X<256 THEN 65
3Ø REM ako je X veće od 255 postaviti MSBX bit
35 POKE 53264, PEEK (53264) OR 2↑A
4Ø X=X—256
45 POKE 53248 + A*2,X
5Ø REM pohranjivanje X-koordinate
55 RETURN
6Ø REM ako je X manje od 256 postavite MSBX bit
65 POKE 53248 + A*2,X
75 POKE 53264, PEEK (53264) AND NOT (2↑A)
8Ø RETURN
```

Proširenje spriteova

Osnovna veličina spriteova većinom zadovoljava, no ponekad im treba dvostruko povećati veličinu. Sprite se može povećati u oba smjera (X i Y), odnosno sa 24 × 21 točke na 48 × 42 točke. To se postiže množenjem bitova u tzv. Sprite Expansion registru. X-registar za proširenje nalazi se na lokaciji 53277, a Y na lokaciji 53271. Kao i prije bit Ø kontrolira sprite Ø, bit 1 — sprite 1, itd. Na primjer, ako želimo prikaz spritea 3 proširenog u X-osi, utipkamo:

```
POKE 53277, PEEK (53277) OR 8
```

Sudari spriteova

U animaciji često treba izvesti sudare između spriteova ili sudar spritea s pozadinom. Broj sudara može se otkriti PEEK-iranjem para registara.

Lokacija za sudar dvaju spriteova je 53278, a za sudar spritea i pozadine 53279.

U svakom registru bit Ø označuje sprite Ø, bit 1 — sprite 1, itd. Glavna prednost registara za sudare je u njihovu automatskom resetu (nakon PEEK), pa programer ne mora brinuti oko resetiranja. Uz sudar (koliziju), podudaranje je vrlo značajan prioritet spriteova koji označuje koji je od dva spritea »prednji«, odnosno koji se nalazi ispred drugog, ili da li je sprite ispred ili na pozadini.

Pri promatranju prioriteta spritea prema pozadini vidljivo je da se prioritet odabire individualno za svaki sprite, odnosno postavljanjem odgovarajućeg bita u Sprite Background Priority registar. Taj registar je smješten na adresi 53275. Ponovno imamo primjer da bit Ø kontrolira sprite Ø, bit 1 — sprite 1, itd. Ako je odgovarajući bit = Ø, sprite će prolaziti ispred pozadine, a ako je 1, onda će prolaziti iza pozadinske grafike. Primjerom se može pokazati kako sprite 4 prolazi iza pozadinske grafike:

POKE 53275, PEEK (53275) OR 16

GRAFIKA VISOKE RAZLUČIVOSTI — GRAFIKA EKRANA, MEMORIJSKA I KOLOR-PODJELA

Grafika visoke razlučivosti prilična je zagonetka jer osnovni priručnik ne govori ništa o njoj. Upotrebljavati grafiku visoke razlučivosti nije lako, a ovdje ćemo iznijeti neke osnovne elemente.

Ekranski prikaz visoke razlučivosti na računalu C64 obuhvaća 64000 individualnih grafičkih točaka (320 × 200 točaka) —320 horizontalno i 200 točaka vertikalno. Dakle, postoji 64000 mjesta za adresiranje točaka unutar memorije RAM. Svaku točku kontrolira bit memorije. Ako je bit = 1 točka svijetli, a ako je Ø onda ne svijetli. Svaki byte u RAM-u kontrolira 8 točaka pa je potrebno 8000 byteova za pohranjivanje cjelokupnog ekranskog prikaza. U 64K RAM-a postoji 8 različitih mjesta gdje se može smjestiti ekranski prikaz grafike visoke razlučivosti (65536 : 8000 = 8,192).

U BASIC-u iskoristive su samo prve dvije lokacije. Razlog leži u čipu VIC koji kontrolira ekran i može iskoristiti samo prvih 16K RAM-a. Dvije moguće lokacije startaju od adrese Ø i od adrese 8192. Međutim jedino moguće mjesto za lociranje ekrana s grafikom visoke razlučivosti je od adrese 8192 do 16192, i to zato što operacijski sustav zauzima niže memorijske lokacije. Donji dio BASIC-a mora se podići na adresu

16193. Moguće je podići i na adrese 16384. Dno BASIC-a može se pomicati samo u blokovima od po 256 bytea.

Primjer:

```
NEW
POKE 642,64 : POKE 44,64 : POKE 46,64 :
POKE 48,64 : POKE 5Ø,64 : POKE 16384,Ø
NEW
```

Rezultat premještanja BASIC-a je gubitak 14K programskog prostora. Ako nam u programu treba veći broj spriteova npr. njih 32 mogu se njihovi podaci spremiti na adrese 2048 do 4096. Od 4096 do 8192 nije ih moguće spremati jer čip VIC podržava na tim adresama set znakova iz ROM-a. Kao što je spomenuto, dio čipa VIC koji kontrolira ekran s »visokom grafikom« i otkriva gdje je ekranski prikaz (screen) smješten izgleda kao bit 3 u VIC Memory Control registru lociranom na adresi 53272. Ako je sadržaj bita Ø, ekranski prikaz počinje na adresi ØØØØ, a ako je sadržaj 1 prikaz počinje na adresi 8192. To znači da uvijek moramo postaviti sadržaj bita ako radimo s BASIC-om. Na primjer, za lociranje prikaza u visokoj grafici na adresi 8192:

```
POKE 53272, PEEK (53272) OR 8
```

Sada je potrebno čipu VIC reći da uključi mod s grafikom visoke razlučivosti. To se postiže postavljanjem bita 5 u VIC kontrolnom registru koji je na adresi 53265. Ako je bit=1 tada je ekranski prikaz u »visokoj grafici«, a ako je bit=Ø tada je u normalnom tekst-modu.

Na primjer:

```
POKE 53265, PEEK (53265) OR 32
```

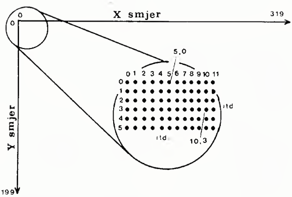
Čip VIC sada prestaje prikazivanje u tekst-modu te uključuje grafiku visoke razlučivosti. Tu se mogu javljati razni problemi jer npr. ekran može biti pun nepoželjnog sadržaja. Prvo, memorijske lokacije od 8192 do 16192 mogu proizvesti crvene točke na plavoj pozadini, ili drugo, svaki znak koji je bio prikazan u normalnom tekst-modu može biti prikazan kao različito obojene točke. To se događa zato što je dio memorije za tekstualni prikaz na ekranu iskorišten za pohranjivanje informacija o boji.

Prije nego što se počne s ekranskim prikazom treba »očistiti« ekran. Obojeni kvadratići mogu se lako maknuti printanjem znakova »clear screen« (SHIFT/CLR) ili rutinom u BASIC-u:

```
5 FOR A=Ø TO 7999
15 POKE 8192+A,Ø
25 NEXT A
```

Organizacija ekrana i uporaba

Već je spomenuto da je ekranski prikaz u visokoj razlučivosti organiziran sa 320 točaka u svakom od 200 redova. U gornjem lijevom kutu je ishodište, dakle točka Ø,Ø. U X-smjeru točke idu od Ø do 319, a u Y-smjeru od Ø do 199.



Ekranski prikaz u grafici visoke razlučivosti sastavljen je od 320 × 200 točkica

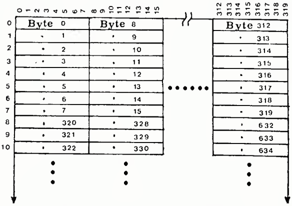
Unutar memorije postoji tzv. stvaran (fizički) izgled ekranskog prikaza. Usporedivši slike može se vidjeti da koordinata 5,Ø odgovara bitu 2 od bytea Ø. Isto tako npr. koordinata 1Ø,3 odgovara bitu 5 od bytea 11.

Numeriranje koordinata je potrebno pri crtanju u grafici visoke razlučivosti. To znači da je potrebna rutina za pretvorbu koordinata u stvarno (fizičko) numeriranje ekranskog prikaza s obzirom na memoriju.

Rutina navedena kao primjer izvodi ovu pretvorbu, odnosno koordinate X i Y daje kao varijable koje će se pretvoriti u stvarne byteove ekranskog prikaza te dati odgovarajući bit byteu da točka svijetli na ekranu:

```
 5 REM rutina za pretvorbu
15 A=8192 + INT(Y/8)*32Ø +
   INT(X/8)*8 + (Y AND 7)
25 POKE A. PEEK (A) OR 2↑ (7—(X AND 7))
```

Izražavati se u grafici visoke razlučivosti znači crtati točke ili linije koje su zapravo sastavljene od točaka. Za primjer može se navesti rutina koja crta liniju između varijabli (točaka) A



Ova slika prikazuje stvaran (»fizički«) izgled ekranskog prikaza u grafici visoke razlučivosti

i B, te točaka C i D. Na kraju rutine sadržaj A i B poprima vrijednosti C i D. To može biti korisno npr. pri crtanju kvadrata.

```
  5 M= SGN(C-A) : P= SGN(D-B)
 15 J= ABS (C-A) : K= ABS(D-B)
 25 Z=K
 35 REM crtanje linije ponavljanjem točaka
 45 IF Z>Ø THEN Z=Z-J : B=B+P
 55 IF Z<=Ø THEN Z=Z+K : A=A+M
 65 GOSUB 115
 75 IF A<>C OR B<>D THEN 45
 85 RETURN
 95 REM rutina za PLOT A, B
115 X=8192+INT(B/8)*32Ø +
    INT(A/8)*8 + (B AND 7)
125 POKE X, PEEK (X) OR 2↑ (7—(A AND 7))
135 RETURN
```

Ako su pri crtanju potrebne kružnice, mogu se crtati slijedećom rutinom. Središte označimo kao koordinate S i C, a radijus s R:

```
  5 A=S+R : B=C
 15 GOSUB 115
 25 REM crtanje ostatka kruga
 35 FOR D=Ø TO 36Ø STEP 1Ø
 45 NX=INT(R*COS(A*Π/18Ø)+S)
 55 NY=INT(R*SIN(A*Π/18Ø)+C)
 65 GOSUB 95
 75 NEXT D
 85 END
 95 REM upotrijebite rutinu za crtanje linija
    iz prethodnog primjera — prilagodite va-
    rijable ovoj rutini
115 REM upotrijebite rutinu za crtanje točke
    (rutina za pretvorbu) — prilagodite vari-
    jable ovoj rutini
```

Primjena boje

Podaci o boji za prikaz ekrana u visokoj razlučivosti dolaze iz područja za normalni prikaz teksta (adrese 1024 do 2023). Svaka »text-screen« lokacija tvori informaciju o boji — kvadrata 8 × 8 točaka pri visokoj razlučivosti. Byte na adresi 1024 tvori kolor-informaciju za bytove 8 do 15 itd.

Svaki kolor-byte ima 4 bita višeg reda (bitovi 4 do 7) za svaki bit pri prikazu visoke razlučivosti koji je postavljen kao 1 i 4 bita nižeg reda (bitovi Ø do 3) za bitove koji su postavljeni na nulu (Ø). To znači da »kolor-memorija« ima nižu razlučivost nego memorija ekrana, pa stoga možemo obojiti ekran samo u blokovima od 64 bita (8 × 8 točaka).

Kao i u bojenju spriteova, ekranski prikaz grafike visoke razlučivosti može operirati na dva načina:

1. Standardni kolor-mod — tu svaka točka može biti u jednoj od dvije boje. Te se boje uzimaju iz bitova višeg i nižeg reda — iz memorije »text-screen«. Dakle, ukupna vrijednost može bi-

ti upisana u cijeloj memoriji za prikaz teksta na ekranu. Rutina u BASIC-u prikazuje kako se može upotrijebiti prikaz u grafici visoke razlučivosti uz boju prednje strane (varijabla BI) i boju pozadine (BP):

```
 5 A = BI*16 + (BP AND 15)
15 FOR C = Ø TO 999
25 POKE 1Ø24 + C, A
35 NEXT C
```

2. Prošireni (extended) kolor-mod — u tom modu na uštrb razlučivosti povećavamo mogući broj upotrijebljenih boja. Vodoravne točkice sada se očitavaju kao parovi, s time da svaki par može biti u jednoj od četiri boje. Kako se parovi kodiraju u boje može se vidjeti iz tablice:

BOJA UZETA SA:	PAR TOCAKA
— kolor-registra pozadine Ø	ØØ
— 4 bita višeg reda iz memorije »text-screen«	Ø1
— 4 bita nižeg reda iz memorije »text-screen«	1Ø
— normalna kolor-memorija	11

Kolor-registar pozadine Ø smješten je na adresi 53281 i podržava boju pozadine. Normalna kolor-memorija je smještena na adresama 55296 do 56295 te se upotrebljava za boju znakova ili u visokoj grafici kao kvadrat od 8 × 8 točaka. Na primjer, adresa 55296 podržava boju svakog para točaka postavljenih na 11 pri byteovima Ø do 7 u ekranskom prikazu u visokoj razlučivosti. Lokacija 55297 podržava par točaka postavljenih na 11 kod byteova 8 do 15 itd.

Za određivanje proširenog kolor-moda za prikaz u grafici visoke razlučivosti treba postaviti bit 4 drugog VIC kontrolnog registra koji je lociran na adresi 53270:

```
POKE 5327Ø, PEEK (5327Ø) OR 16
```

Za detaljno proučavanje grafike visoke razlučivosti treba koristiti mnogo dopunske literature, a za uspješno crtanje upotrebljavati Simon's BASIC ili neke od dobrih profesionalnih programa za crtanje npr.: Super Sketch, Koala Pad, Doodle, Paint Magic itd.

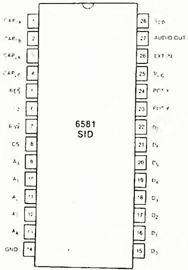
Naravno poželjna je i upotreba hardverskih dodataka kao npr. svjetlosne olovke (light-pen), digitalizirajuće ploče (digitazing-pad) itd.

ZVUK NA C 64

ZVUK I ZVUČNI EFEKTI

Računala mogu učiniti gotovo sve što zamislite, pa i više od toga, uz dovoljan utrošak znanja i vremena. Mogu voditi vaše kućanstvo, animirati TV-slike tako da se ne razlikuju od realnosti, a mogu i svirati. Dosad ste vjerojatno mnogo puta čuli muziku generiranu pomoću računala, a da toga niste ni bili svjesni. Razlog je u tome što su stvoreni kompjuteri i programi koji omogućavaju imitiranje standardnih instrumenata do u najmanji detalj. S jedne strane tako su smanjeni troškovi orkestra, a s druge kompozitor dobiva potpunu kontrolu nad muzikom koju stvara. Većinom kompjuterski generirani zvukovi služe kao muzička pratnja ili glazbena kulisa u filmu ili na televiziji. Kompjuter je idealan za mehaničko izvođenje glazbe, jer pruža bezbroj mogućnosti manipulacije s osnovnim parametrima od kojih se zvuk sastoji. Naravno, ne postoji razlog zbog kojeg bi takva muzika morala zvučiti baš kao da je iz kompjutera. Kada čujete kompjutersku muziku koja cvili, pišti i zuji poput »raštimanog« radio-aparata, znajte da je to samo zato jer je kompozitor/programer želio takav zvuk. Kompjuter nije ograničen samo na takve zvukove, već može bez problema izvoditi dobar jazz, pop, rock ili čak klasičnu glazbu. A da bi kompjuter mogao uopće svirati, potrebno je prvo definirati muzički model koji će nam omogućiti da se što više muzičkog »znanja« kondenzira i spremi kao niz brojeva u memoriju

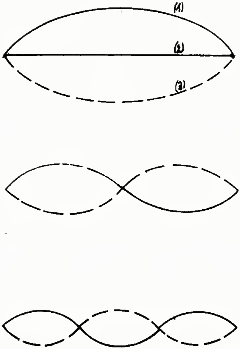
kompjutera. Takav model, srećom, već postoji i ugrađen je u vaš COMMODORE 64. Tako je sviranje pomoću računala znatno olakšano te zahtijeva samo bazično razumijevanje zvuka i muzike da biste mogli iz njega izvući što više. U C-64 kontrolu zvuka nosi poseban čip, nazvan SID (SOUND INTERFACE DEVICE), koji programirate relativno jednostavnim komandama iz BASIC-a. Naravno, ako ne želite sami programirati svoja muzička djela, na raspolaganju vam stoje deseci komercijalno razvijenih programa, koji rade sav mukotrpni posao i vama ostavljaju slobodne ruke za skladanje i sviranje. Ti muzički editori identični su programima za obradu teksta te pružaju punu kontrolu nad muzičkim materijalom, a njima se služe i profesionalni muzičari. Ipak, koliko god ti programi bili korisni, potrebno je razumijevanje potencijala čipa SID da bi se mogle u potpunosti iskoristiti sve mogućnosti. A one su gotovo beskrajne. Ako istražimo sve kontrole čipa SID (koje ćemo kasnije detaljnije objasniti), jednostavnim matematičkim postupkom dolazimo do više od četiri (4!) milijarde različitih kombinacija zvuka za svaku pojedinačnu notu. Očito je da se neke od tih kombinacija neće mnogo razlikovati od drugih te da će neke zvučati užasno, no broj korisnih je ipak toliko velik da je zaista potrebno razumijevanje postupka kojim kompjuter stvara muziku. Tako ćete se ipak moći nekako snaći i moći birati određene tonove u tom mnoštvu, a nećete gotovo



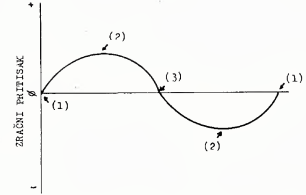
uzaludno pokušavati s dobrom starom metodom pokušaj-pogreška.

GENERIRANJE ZVUKA

Zamislite nit nategnutu između dvije točke. Ako je gurnemo ili povučemo sa strane, ona će



zatitrati kao na slici, polako smanjujući put kretanja, sve dok se ne zaustavi. Dok se pomiče, nit pokreće zrak i uzrokuje zračni pritisak koji



se u uhu percipira kao zvuk. Promjena zračnog pritiska na bubnjiću uha grafički je prikazana na slici. To je poznati sinusoidalni oblik (tako nazvan, jer se matematički može izračunati pomoću sinus-funkcije).

Brzina kojom nit titra ovisi o njenoj dužini i zategnutosti te može varirati od nekoliko titraja do više tisuća titraja u sekundi. Ako je frekvencija titranja ispod otprilike 50 titraja ili iznad 20000 titraja u sekundi, ljudsko uho više neće moći osjetiti te promjene zračnog pritiska kao zvuk. Pri skraćivanju niti, ona će titrati brže i proizvoditi će viši ton, a pri produženju će usporiti i emitirati niži ton, a o mjestu na koje-

mu ćete žicu potaknuti na titranje, ovisi njen zvuk. Slike b i c nam pokazuju što se dešava. Nezavisno od glavnog titranja, dvije polovice niti mogu također vibrirati svaka emitirajući istu notu. Frekvencija nove note, poznate kao drugi harmonik, je točno dva puta veća od osnovne note, nazvane prvi harmonik.

Ako pođemo dalje, nit može vibrirati na svojim trećinama, četvrtinama, petinama, šestinama itd. Praktično govoreći, jedna nit može simultano proizvesti svih 12 nota koje imamo u našem muzičkom sistemu, no one će se nalaziti u višim oktavama od osnovne.

Svaki harmonik je tiši od prethodnoga i zbog toga viši iščezavaju prije nižih dok nit usporava gibanje.

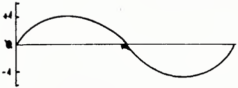
Bogatstvo zvuka koje uho čuje iz klavira u usporedbi sa sinusoidalnim valom iste frekvencije, potječe od mješavine mnogih nota uzrokovanih kompleksnim titranjem žice. Zvuk gitare je drukčiji, jer titranje kraćih žica producira harmonike različitih jakosti nego udar po dugoj žici s batićem u klaviru.

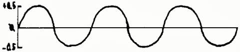
Elektronički je vrlo lako kreirati sinusni val pomoću analognog sklopa koji direktno kontrolira oscilator (varijacijama u naponu), ili digitalno uz brojeve dobivene iz grafa kao na slici i pohranjene u memoriju kompjutera. Sklop koji pretvara brojeve u napon poznat je pod imenom digitalno-analogni (DA) konvertor.

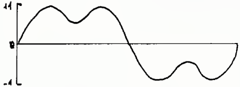
COMMODORE 64 ima 3 oscilatora (kanala), a čak i profesionalni instrumenti rijetko pružaju više od 16 kanala.

Aditivna sinteza

Digitalna tehnika je omogućila biranje pojedinih harmonika i njihovo zbrajanje da bi se dobio kompleksni signal koji slušatelj percipira kao mješavinu jednostavnih sinusnih valova. Primjer tog načina sinteze (aditivne — zbrajajuće) vidi se na slici. U gornjem dijelu imamo dva sinusna vala koji označavaju izlaz dva odvojena oscilatora. Kao što vidimo, amplituda na y-osi prvog vala je 1, a drugog 0.5. Kako su to samo relativni odnosi, shvatimo prvi val kao 100%, a drugi 50% njegove vrijednosti. Val jedan je osnovni, a val dva je viši harmonik. Sada pretpostavimo da umjesto dva imamo samo jedan oscilator. Možemo se zapitati da li je zračni pritisak koji dolazi do našeg bubnjića zbroj pritisaka oba oscilatora. Odgovor je potvrdan i zbog toga možemo jednostavno zbrojiti dva vala da dobijemo val tri, koji će stvarati identičan zračni pritisak kao ta dva, ali će se emitirati samo s jednoga oscilatora. Valni oblik 3 dobili smo običnim grafičkim zbrajanjem dva vala po amplitudi i vremenu. Rezultat izgleda kao sinusoida s izbočenjima i udubljenjima. Spajanje zajedno sinusnih valova viših harmonika, negdje do 25., omogućava kompjuteru da vjerno imitira neke od kompleksnih muzičkih zvukova, kao

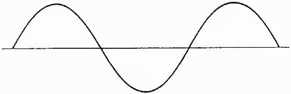


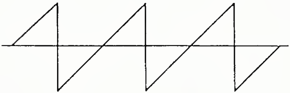


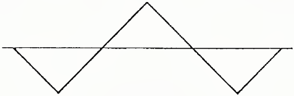


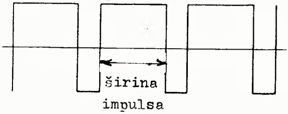
npr. trube ili ksilofona. S povećanjem broja kanala (oscilatora) rastu i mogućnosti samog sistema.

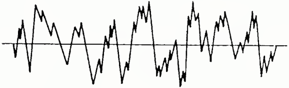
Čip SID koji je ugrađen u COMMODORE 64 ima programirana četiri valna oblika (slika) u svoj ROM, a jedan od njih može se i varirati u određenim granicama. Ti valni oblici su oblik











pile, koji služi za simuliranje žičanih instrumenata, oblik trokuta, koji stvara zvuk nalik nekim duhačkim instrumentima, pravokutni oblik, koji korisnik može kontrolirati, i šum koji služi za imitaciju perkusionih instrumenata (udaraljki).

Ti su oblici zajednički za svu kompjutersku muziku pošto su matematički dobro definirani.

Pilasti oblik i trokutni val dobivaju se također aditivnom sintezom.

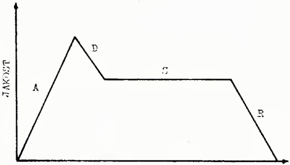
Šum se sastoji od slučajno biranih točaka i ne može se sintetizirati iz seta harmonika.

Još nam preostaje kvadratni (pulsni) oblik, koji je vrlo zanimljiv zbog nekoliko pojedinosti. Kao prvo, to je valni oblik koji je potpuno elektronski i novi je zvuk koji ne postoji u prirodi. Ne dobiva se zbrajanjem prirodnih harmonika (iako je i to moguće) već se stvara uključivanjem i isključivanjem određenog sklopa. Kako se takav sklop može jednostavno dizajnirati i izraditi, pulsni valni oblik se najčešće sreće u sintetiziranoj muzici. No, čip SID posjeduje karakteristiku koja nije toliko česta. Korisnik može birati širinu impulsa i time kontrolirati valni oblik. Budući da je za kontrolu širine impulsa odvojeno 12 bita, to znači da je moguće stvoriti čak 4096 raznih oblika.

ZVUČNE OVOJNICE (ENVELOPE)

Razliku u zvuku instrumenata ne stvaraju samo razlike u valnim oblicima, već i način na koji su žice potaknute na titranje. Upravo zbog toga se iskorištava i drugi (jednako važan kao i prvi) dio muzičkog modela, koji kontrolira nastanak note, njen razvoj do pune jakosti i njeno nestajanje.

Takav muzički uzrok je modeliran u čipu SID dijeljenjem svake note na četiri dijela, od kojih ne moraju svi biti upotrijebljeni za svaki instrument.



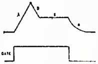
Prvi dio je vrijeme porasta (engl. ATTACK) u kojemu nota raste od nule do svoje maksimalne jakosti. Drugi dio (DECAY), vrijeme pada, je pad s inicijalne jakosti na treći dio, održavanje (SUSTAIN), koji može trajati neodređeno vrijeme. Četvrti dio, opuštanje (RELEASE), upravlja padom jakosti od nivoa održavanja do potpune tišine. Kako ta četiri stupnja zatvaraju valnu formu i ograničavaju njenu amplitudu, upotrebljava se izraz OVOJNICA (ENVELOPE). Najčešće se kao kratica uzima ime ovojnica ADSR, a čine je početna slova engleskih naziva stupnjeva (Attack, Decay, Sustain, Release).

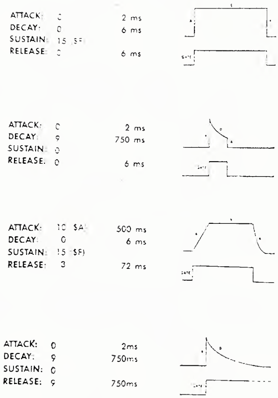
Čip SID omogućava 16 različitih vrijednosti za svaki od ta četiri parametra. Vrijeme porasta (Attack) može varirati od 2 ms do 8 sek, dok parametri D i R mogu trajati od 8 ms do 24 sek. To nam ukupno daje 65536 mogućih ovojnica. Naravno, i tu vrijedi sistem pokušaja i pogrešaka da bi se dobio željeni zvuk. Kao savjet možemo reći da klavir, npr., nema parametar S, ali ima dugačko vrijeme nestajanja ako se tipka drži, ili vrlo brzo nestajanje ako se tipka odmah otpusti. Tada imamo parametre ADR ili AD, dok su i ostali tu, ali jednaki nuli. Za elektronske, nekonvencionalne zvukove, dobro je iskoristiti dugo vrijeme porasta (A). Kako je izbor zaista velik (oko 65000), dobro je zabilježiti parametre koji daju dobre rezultate, za kasniju upotrebu.

FILTERI

Jedna od interesantnih karakteristika čipa SID jest i imitiranje analognog sintesizera pomoću filtera — niskog, visokog te pojasnog propusta koji se dobiva kao kombinacija prva dva. Služe za dodatno korigiranje zvuka koji stvaramo. Uz to, SID pruža mogućnost sinhronizacije, frekventne modulacije i rezonancije. Sinhronizacija omogućuje kontrolu jednog kanala pomo-

ATTACK:	10 ($A)	500 ms
DECAY:	8	300 ms
SUSTAIN:	10 ($A)	
RELEASE:	9	750 ms





ću drugog, te sprečava preklapanje nota i kanala. Frekvencijska modulacija omogućuje da se kanalom 3 kontroliraju prvi ili drugi kanal. Napokon, SID omogućuje i 16 stupnjeva rezonancije vraćanjem nekog od filtriranih zvukova kroz filter s malim zakašnjenjem.

Pa, da zbrojimo sve mogućnosti koje smo spomenuli i pokušamo nakon toga objasniti naredbe kojima se kontrolira čip SID. Imali smo 4 ovojnice sa 16 mogućih vrijednosti, 4 valne forme, od kojih jedna ima 4096 potformi, dva filtera, jedan sa 8 vrijednosti i drugi sa 256, te zatim sinhronizaciju, FM i rezonanciju. Ako sve to pomnožimo da dobijemo broj kombinacija, rezultat je približno 4.40126×10^{12} kombinacija. Broj je impresivan, no SID je ipak limitiran sa svoja 3 kanala i unaprijed određenim valnim oblicima. Za usporedbu možemo samo spomenuti da neki, istina malo skuplji modeli računala, omogućavaju proširenje na 16 kanala i kreiranje svake forme specificiranjem do 24 harmonika, a uz to stvaraju i stereozvuk. Računanje broja kombinacija takvih sistema prepuštamo vama, ako ste raspoloženi za takav posao.

PROGRAMIRANJE ZVUKA NA C64

Podaci kojima ćemo se služiti pri muzičkoj sintezi nalaze se u COMMODORE-ovoj memoriji na adresama od 54272 do 54296. Osnovna adresa koju bi trebalo zapamtiti jest 54272, a ostale

se adrese definiraju kao pribrojnici tom broju (54272 + (Ø—24)). U te ćemo adrese upisivati podatke naredbom POKE adresa, pod. Varijable koje upisujemo smiju imati vrijednost od Ø do 255.

Počet ćemo prvo s frekvencijom željenoga zvuka koju upisujemo u prva dva registra čipa SID, tj. na adrese 54272 (nazovimo je SI) i SI + 1. No, da bismo mogli upisati broj koji označava frekvenciju u ta dva byta, prvo je moramo konvertirati u oblik shvatljiv računalu. Naime, frekvencija se upisuje u spomenuta 2 bytea, koji se zovu »niži« (SI) i »viši« (SI + 1). U priloženoj tablici ćete naći neke od vrijednosti »višeg« i »nižeg« bytea za određene frekvencije, no ako želite sami izračunati neku međuvrijednost, služite se formulama. Prvo izaberite frekvenciju Fizl. Zatim izračunajte vrijednost Fn = Fizl//0.06097, te vrijednost zaokružite na cijeli broj izostavljajući decimale iza nje. Da biste dobili vrijednosti byteova, izračunajte:

Fvis = INT(Fn/256)

Fnis = Fn—(256✳Fvis)

te ih naredbom POKE upišite u odgovarajuće lokacije: SI (Fnis) i SI + 1 (Fvis).

TABLICA NEKIH FREKVENCIJA
I VRIJEDNOSTI »VIŠEG« I »NIŽEG« BYTEA

Frekvencija Fizl (Hz):

Fizl	2145	2551	3034	5728	6812	7647	8583
Fvis	8	9	11	22	26	29	33
Fnis	97	247	218	96	156	223	135

Fizl	9094	9634	10814	11457	12860	14435
Fvis	35	37	42	44	50	56
Fnis	134	162	62	193	60	99

Da biste kontrolirali glasnoću zvuka, trebate upisati u lokaciju SI+24 vrijednosti od Ø do 15, naredbom POKE SI+24,X — (X=Ø—15, SI=54272).

Adresa SI+4 služi nam da bismo izabrali valnu formu kojom ćemo generirati zvuk. Kao što smo spomenuli, postoje 3 valne forme i šum, a izabiremo ih tako da u lokaciju SI+4 upisujemo vrijednosti 17 (trokutni signal), 33 (pilasti), 65 (pravokutni) ili 129 (šum). Za kanal 1 to je adresa SI+4, za kanal 2 to je SI+11 i za kanal 3 to je SI+18.

kanal	1	2	3	šum	kvadrat	trokut	pila
regist.	4	11	18	129	65	17	33

Da bismo isključili valni oblik, tada u odgovarajući registar upišemo vrijednost (17,33,65, 129)—1, tj. vrijednost (16,32,64 ili 128).

Kao što smo prije spomenuli, jedan od tih valnih oblika, četvrtasti, ima mogućnost mijenjanja dužine impulsa. Da bismo ga kontrolirali koristimo registre:

SI+2 i SI+3.

Registar SI+2 je »niži«, a SI+3 »viši« byte. U niži se upisuje broj (Nk) od 0 do 255, a u viši (Vk) od 0 do 15. Ako slučajno izaberemo te bro-

jeve. slijedeća formula nam daje širinu impulsa u postocima koju ćemo dobiti na izlazu:

Pv=Vk✳256+Nk
Pizl=(Pv/40.95)%

Npr., za vrijednost Pv=2048 treba registre namjestiti na Vk=8 i Nk=0.

Kao demonstracija dosad spomenutih podataka može poslužiti ovaj program:

```
  5 SI=54272
 1Ø FOR LI=SI TO SI+24:POKE LI,Ø:NEXT
 15 POKE SI+3,4:POKE SI+2,16
 2Ø POKE SI+5,9:POKE SI+6,Ø
 3Ø POKE SI+24,15
 4Ø READ VF,NF,P
 5Ø IF VF<Ø THEN END
 6Ø POKE SI+1,VF:POKE SI,NF
 7Ø POKE SI+4,17
 8Ø FOR F=1 TO P:NEXT
 9Ø POKE SI+4,16:FOR F=1 TO 5Ø:NEXT
1ØØ GOTO 4Ø
11Ø DATA 25,177,25Ø,28,214,25Ø
12Ø DATA 25,177,25Ø,25,177,25Ø
13Ø DATA 25,177,125,28,214,125
14Ø DATA 32,94,75Ø,25,177,25Ø
15Ø DATA 28,214,25Ø,19,63,25Ø
16Ø DATA 19,63,25Ø,19,63,25Ø
17Ø DATA 21,154,63,24,63,63
18Ø DATA 25,177,25Ø,24,63,125
19Ø DATA 19,63,25Ø,—1,—1,—1
```

Ako u ovom programu promijenite liniju 15 u: 15 POKE SI+3,8:POKE SI+2,Ø, te u liniji 7Ø promijenite vrijednost 17 u 65 i u liniji 9Ø broj 16 u 64, osjetit ćete razliku. Zatim promijenite SI+3,8 u SI+3,1 i ponovno poslušajte.

Za kontrolu parametara ADSR služe registri 5 i 6, tj. SI+5 i SI+6. Kako se za 4 parametra troše 2 byta, oni se dijele na polovice od 4 bita, tj. »više« ili »niže«. Parametar A (vrijeme porasta) je viša polovica registra 5 i njegove vrijednosti su Ø do 15, te se one množe s brojem 16 i zbrajaju s vrijednošću 4 niža bita (Ø—15), koji kontroliraju parametar D ovojnice. Taj broj koji dobijemo upisujemo u registar SI+5. Isto tako se izračunavaju i parametri S i R, koji se nalaze u registru SI+6. S(ustain) je viši, a R(elease) niži dio tog bytea.

U tablici je pregled trajanja tih parametara u odnosu na unesene vrijednosti:

vrijednost:	Ø	1	2	3	4	5	6	7	8
A(ttack)	2ms	8ms	48	72	114	168	204	240	300
D/R	6ms	24	16	24	38	56	68	80	100

vrijednost:	9	10	11	12	13	14	15
A(ttack)	750	1.5s	800ms	3s	9s	15s	24s
D/R	250	500ms	2.4s				

Kao primjer, promijenite liniju 2Ø u programu u POKE SI+5,88:POKE SI+6,89 ili

```
2Ø POKE SI+5,9:POKE SI+6,9
7Ø POKE SI+4,17
9Ø POKE SI+4,16:FOR T=1 TO 5Ø:NEXT
```

ili

```
2Ø POKE SI+5,144:POKE SI+6,243
```

Raspored kontrola ovojnica za pojedine kanale je ovakav:

kanal	1	2	3	4 bita & 4 bita
AD reg	5	12	19	(Ø—15)✕16 + (Ø—15)
SR reg	6	13	20	(Ø—15)✕16 + (Ø—15)

Ako želite upotrijebiti filtere, uzet ćemo registar 23 naredbom POKE SI+23,1 koja uključuje filter i registar 24, koji inače služi i za kontrolu glasnoće (njegova 4 niža bita — vrijednosti od Ø do 15. Od 4 viša bita registra 24, služimo se sa 3 da bismo izabrali vrstu filtera — visoki, niski ili pojasni. Bit 6 kontrolira filter visokog propusta, a da bismo ga aktivirali trebamo u registar SI+24 upisati vrijednost glasnoće (Ø—15)+64. Za pojasni filter služi bit 5, a aktivira se sa POKE SI+24, (glasnoća)+32. Filter niskog propusta aktiviramo sa POKE SI+24, (glasnoća)+16. Svi se ovi filteri deaktiviraju upisivanjem samo iznosa glasnoće (Ø—15).

Uz filtere moramo izabrati i frekvencije na kojima ćemo rezati, a upisujemo ih u registar 21 (donja 3 bita: vrijednosti Ø—7) i registar 22 (8 bita: vrijednosti Ø—255). Pravilnom upotrebom filtera možemo dobiti vrlo zanimljive zvučne efekte.

Uz ostalo, čip SID omogućuje i kreiranje kompleksnih harmoničkih struktura pomoću sinhronizacije i kružne modulacije dvaju kanala. Sam proces sinhronizacije je logičko povezivanje (AND) dviju valnih formi. Ako je jedna od njih jednaka nuli, izlaz je nula.

Primjer povezivanja kanala:

```
 1Ø SI=54272
 2Ø FOR LI=Ø TO 24:POKE SI+LI,Ø:NEXT
 3Ø POKE SI+1,1ØØ
 4Ø POKE SI+5,219
 5Ø POKE SI+15,28
 6Ø POKE SI+24,15
 7Ø POKE SI+4,19
 8Ø FOR F=1 TO 25ØØ :NEXT
 9Ø POKE SI+4,18
1ØØ FOR F=1 TO 1ØØØ :NEXT:POKE
    SI+24,Ø
```

Kako vidimo, u liniji 7Ø je za trokutni valni oblik sinhronizacija postignuta setiranjem bita 1 (vrijednost 2), čija se vrijednost pribraja broju 17 koji normalno određuje trokutni val.

Kružna modulacija zamjenjuje trokutni izlaz oscilatora 1, kombinacijom oscilatora 1 i 3, a aktivira se bitom 3 četvrtog registra.

Da bi se imao lakši pregled registara i njihovih funkcija, ovdje je tablica koja sadržava te podatke:

REGISTAR	ADRESA	FUNKCIJA
Ø	54272	Kanal 1, frekvencija, niži byte
1	54273	Kanal 1, frekvencija, viši byte

2	54274	Kanal 1, širina pulsnog signala, niži byte
3	54275	Kanal 1, širina pulsnog signala, viši byte
4	54276	Kanal 1, kontrola v. oblika
5	54277	Kanal 1, A/D kontrola
6	54278	Kanal 1, S/R kontrola
7	54279	Kanal 2, frekv., niži byte
8	5428Ø	Kanal 2, frekv., viši byte
9	54281	Kanal 2, širina puls. sig., niži byte
10	54282	Kanal 2, širina puls. sig., viši byte
11	54283	Kanal 2, kontrola v. oblika
12	54284	Kanal 2, A/D kontrola
13	54285	Kanal 2, S/R kontrola
14	54286	Kanal 3, frekv., niži byte
15	54287	Kanal 3, frekv., viši byte
16	54288	Kanal 3, šir. puls. sign., niži byte
17	54289	Kanal 3, šir. puls. sign., viši byte
18	5429Ø	Kanal 3, kontrola v. oblika
19	54291	Kanal 3, A/D kontrola
2Ø	54292	Kanal 3, S/R kontrola
21	54293	Frekv. rezanja filtera, niži byte
22	54294	Frekv. rezanja filtera, viši byte
23	54295	Kontrola filtera
24	54296	Izbor filtera/Jakost tona
25	54297	Potenciometar X — za upotrebu analogne palice za igru
26	54298	Potenciometar Y — za upotrebu analogne palice za igru
27	54299	Izlazni signal, Kanal 3
28	543ØØ	Signal ovojnice, Kanal 3

Ovime nisu nabrojene sve muzičke mogućnosti COMMODORE-a 64, no za to bi bila potrebna cijela knjiga, pa vam preporučujemo da potražite takva izdanja, ako ste posebno zainteresirani za muziku na C64.

DEMONSTRACIJSKI PROGRAMI ZA TVORBU ZVUČNIH EFEKATA

Programi demonstriraju mogućnosti čipa SID i njegove kontrole. Trebate ih upisati i startati naredbom RUN. Nakon što ste čuli zvučni efekt, preporučujemo da pokušate mijenjati vrijednosti u registrima čipa SID, da biste upoznali razlike koje nastaju. Vrijednosti koje se mogu upisivati su naznačene u tablicama ili u samom tekstu ovog poglavlja.

```
READY.

95 REM************************************************************
100 REM DEMONSTRACIJA EFEKTA SIRINE PULSNOG(KVADRATNOG) VALA
105 REM************************************************************
110 DIM FL(8),FH(8)
112 REM*********************************************
115 REM UCITAVANJE PODATAKA O FREKVENCIJI TONOVA
117 REM*********************************************
120 FOR G=1 TO 8:READ FL(G),FH(G):NEXT
130 DATA 195,16,209,18,31,21,96,22
140 DATA 30,25,49,28,165,31,135,33
150 SI=54272
152 REM************************************************
155 REM CISCENJE REGISTARA SID CIPA(PUNE SE S NULOM)
157 REM************************************************
160 FOR J=SI TO SI+24:POKE J,0:NEXT
164 REM************************
165 REM NAMJESTANJE GLASNOCE
166 REM************************
170 POKE SI+24,15
174 REM*****************************
175 REM NAMJESTANJE ADSR PARAMETARA
176 REM*****************************
180 POKE SI+5,54:POKE SI+6,240
190 FOR SV=0 TO 9
200 PRINT "SIRINA KVADRATNOG VALA= ",15*(2↑SV)
210 FOR J=1 TO 8
214 REM**************************
215 REM NAMJESTANJE FREKVENCIJE
216 REM**************************
220 POKE SI,FL(J):POKE SI+1,FH(J)
224 REM*************************************
225 REM NAMJESTANJE SIRINE KVADRATNOG VALA
226 REM*************************************
230 P=15*(2↑SV):POKE SI+3,INT(P/256)
240 POKE SI+2,P-256*INT(P/256)
244 REM**********
245 REM TON ZVUKA
246 REM**********
250 POKE SI+4,65
260 FOR L=1 TO 500:NEXT
264 REM*****************************************
265 REM GENERIRANJE PAUZE(ISKLJUCENJE KANALA)
266 REM*****************************************
270 POKE SI+4,64
280 FOR M=1 TO 50:NEXT
290 NEXT
300 FOR M=1 TO 50:NEXT
310 NEXT

READY.
```

```
READY.

100 REM****************************
101 REM DEMONSTRACIJA EKSPLOZIJE
102 REM****************************
110 SI=54272
120 FOR J=SI TO SI+24:POKE J,0:NEXT
130 POKE SI+24,15
140 FOR G=7 TO 13:POKE SI+5,G:PRINT"G= ",G
150 FOR M=0 TO 15 STEP 3
160 POKE SI,255:POKE SI+1,M
170 POKE SI+4,129
180 FOR H=1 TO 1500:NEXT
190 POKE SI+4,128
200 FOR H=1 TO 50:NEXT
210 NEXT:NEXT

READY.
```

```
READY.

100 REM**************
101 REM ZVUK SIRENE
102 REM**************
110 SI=54272
120 FOR J=SI TO SI+24:POKE J,0:NEXT
130 POKE SI+24,15
140 POKE SI+5,17:POKE SI+6,246
150 FOR L=16 TO 255 STEP 16
160 POKE SI+19,17:POKE SI+20,246
170 POKE SI+14,L:POKE SI+15,0
180 POKE SI+18,17:POKE SI+4,33
190 FOR K=1 TO 100
200 O=8000+4*(PEEK(SI+27))
210 FH=INT(O/256):FL=O-256*FH
220 POKE SI,FL:POKE SI+1,FH
230 NEXT K
240 NEXT L
250 POKE SI+18,16:POKE SI+4,32
260 END

READY.
```

```
100 REM***************************
101 REM DEMONSTRACIJA ZVUKA ZVONA
102 REM***************************
110 DIM FL(8),FH(8)
120 FOR I=1 TO 8:READ FL(I),FH(I):NEXT
130 DATA 135,43,162,37,62,42,193,44
140 DATA 60,50,39,56,75,63,15,67
150 SI=54272
160 FOR J=SI TO SI+24:POKE J,0:NEXT
170 POKE SI+24,15:POKE SI+3,8
180 POKE SI+5,11:POKE SI+6,11
190 FOR A=1 TO 8
200 POKE SI,FL(A):POKE SI+1,FH(A)
209 REM**********************************
210 REM ODABIR IZLAZNOG VALNOG OBLIKA
211 REM**********************************
220 POKE SI+4,65
230 FOR H=1 TO 1000:NEXT
240 POKE SI+4,64
250 FOR H=1 TO 50:NEXT
270 NEXT

READY.
```

```
READY.

100 REM*********************
101 REM DEMONSTRACIJA SUMA
102 REM*********************
110 SI=54272
120 FOR G=SI TO SI+24:POKE G,0:NEXT
130 POKE SI+24,15:POKE SI+6,240
140 POKE SI+5,34
150 FOR M=0 TO 15
160 POKE SI,64:POKE SI+1,M
170 POKE SI+4,129
180 FOR E=1 TO 1500:NEXT
190 POKE SI+4,128
200 FOR E=1 TO 50:NEXT
210 NEXT

READY.
```

```
FOUND BOMB DEMO
LOADING

READY.

100 REM******************
101 REM ZVUK PADA BOMBE
102 REM******************
110 SI=54272
120 FOR J=SI TO SI+24:POKE J,0:NEXT
130 POKE SI+24,15
140 POKE SI+5,17:POKE SI+6,246
150 POKE SI+19,16:POKE SI+20,0
160 POKE SI+14,1:POKE SI+15,4
170 POKE SI+18,33:POKE SI+4,33
180 R=PEEK(SI+28):IF R<240 THEN 180
190 G=24000+64*R
200 FH=INT(G/256):FL=G-256*FH
210 POKE SI,FL:POKE SI+1,FH
220 R=PEEK(SI+28):IF R>32 THEN 190
250 POKE SI+18,16:POKE SI+4,32
260 POKE SI+5,11:POKE SI+6,9
270 POKE SI,1:POKE SI+1,2
280 POKE SI+24,15:POKE SI+4,129
290 FOR H=1 TO 1000:NEXT
300 POKE SI+4,129
310 FOR H=1 TO 1500:NEXT:GOTO 130
320 END

READY.
```

Programi:

- DEMONSTRACIJA EFEKTA ŠIRINE KVADRATNOG VALA
- DEMONSTRACIJA EKSPLOZIJE
- DEMONSTRACIJA SIRENE
- DEMONSTRACIJA ZVUKA ZVONA
- DEMONSTRACIJA ŠUMA
- DEMONSTRACIJA ZVUKA PADA BOMBE

U svim ovim programima registri čipa SID označeni su varijablom SI ili SI+X, gdje je X=(1—28). U zadnja 4 registra podaci se ne mogu upisivati već se samo čitaju naredbom **PEEK** (x).

STROJNI KOD

KARAKTERISTIKE

Vlasnici C64 koji su se već okušali u programiranju uz BASIC uviđaju da njihovi programi zauzimaju puno memorije, rade sporije, da programiranje otežavaju slabosti ugrađenog BASIC-a itd.

Jedan od putova da se troši manje memorije, omogući brži tok programa i izbjegnu manjkavosti BASIC-a, a time naprave i bolji programi jest upotreba tzv. strojnoga kôda (machine code). Program napisan u nešto složenijem strojnom kôdu zauzimat će manje memorije nego isti napisan u BASIC-u, a radit će mnogo brže. Svi programi vrhunske kvalitete izrađeni su u strojnom kôdu.

Za svoju osnovicu strojni kôd ima heksadecimalni brojevni sustav kojemu je baza broj 16. Naviknuti na dekadski brojevni sustav (baza 10) početnici ponekad imaju probleme s upotrebom heksadecimalnog sustava.

Kako strojnim kôdom možemo programirati pridajemo mu obilježje nižeg programskog jezika. Za upis strojnog kôda treba upotrebljavati uvodnike (hex loadere). Pomoću njih računalo zna da ćemo primijeniti strojni kôd.

Prema instrukcijama u BASIC-u, instrukcije strojnog kôda imaju neuobičajen oblik, npr.:

FE 4C
7A 1B itd.

Svaka takva instrukcija ima neko značenje, isto kao instrukcije u jednostavnijem BASIC-u.

Za potpuno savladavanje strojnog kôda bit će potrebna dodatna literatura, jer ova knjižica ne može objasniti sve tajne programiranja u strojnom kôdu.

Vidjevši oblike naredbi možemo se zapitati zašto računalo pomoću njih brže radi i troši manje memorije.

Pisanje programa u strojnom kôdu jest »skraćeno« pisanje jer naredbe su kraće iako istog sadržaja i značenja. Računalo će ih stoga brže obraditi. Manje memorije troši se zato što za formiranje naredbi treba manje byteova memorije. Dakle, u memoriju će stati više instrukcija.

Gore spomenute naredbe zapravo su brojevi u heksadecimalnom obliku. Taj brojevni sustav ima ukupno 16 različitih znamenki (dekadski ima 10 različitih — Ø do 9). U heksadecimalnom sustavu znamenke od Ø do 9 obilježavaju se normalno, a preostalih 6 znamenki slovima od A do F. A je zapravo broj 10, B=11, C=12, D=13, E=14 i F=15.

PRETVORBA HEKSADECIMALNOG U DEKADSKI BROJEVNI SUSTAV

Rekli smo da je baza heksadecimalnog brojevnog sustava broj 16. Znamenke se pišu Ø,1,2, 3,4,5,6,7,8,9,A,B,C,D,E i F, a postoji i razlučni element — zarez.

Dvoznamenkasti brojevi označavaju se slovima radi lakšeg pisanja i prikaza na računalu.

Pogledajmo primjer heksadecimalnog broja: 2C1E

Duže se može pisati kao:

$2 \times 16^3 + C \times 16^2 + 1 \times 16^1 + E \times 16^0$

ili još duže kao:

$2 \times 16^3 + 12 \times 16^2 + 1 \times 16^1 + 14 \times 16^0$

Svaki broj na nultu potenciju ima vrijednost 1 ($16^0 = 1$, $2^0 = 1$, $10^0 = 1$). Kada bismo izračunali gornji izraz ustanovili bismo da broju 2C1E odgovara dekadski broj 11294. Dakle:

$2C1E_{16} = 11294_{10}$

Heksadecimalni brojevni sustav je zapravo pomoć programeru kako ne bi morao neposredno mikroprocesoru utipkavati binarni kôd (nizove nula i jedinica — binarni brojevni sustav).

Ujedno je vrlo pogodan za rad s digitalnim brojevima koji imaju broj znamenaka: 4,8,12,16 itd.

Upravo s brojevima takvog formata rade mikroprocesori pa je heksadecimalni sustav najupotrebljivaniji pri radu s njima.

Kako možemo pretvoriti heksadecimalni broj u nama normalniji dekadski? Ponekad je takva pretvorba potrebna pa je zgodno poslužiti se tablicom:

Heksadecimalna znamenka	POLOŽAJ 3	2	1	0
0	0	0	0	0
1	4096	256	16	1
2	8192	512	32	2
3	12288	768	48	3
4	16384	1024	64	4
5	20480	1280	80	5
6	24576	1536	96	6
7	28672	1792	112	7
8	32768	2048	128	8
9	36864	2304	144	9
A	40960	2560	160	10
B	45056	2816	176	11
C	49152	3072	192	12
D	53248	3328	208	13
E	57344	3584	224	14
F	61440	3840	240	15

Uzmimo broj iz prijašnjeg primjera: $2C1E_{16}$, pa objasnimo kako smo ga dobili iz tablice.

E je nulta znamenka i iz tablice iznosi: 14

1 je prva znamenka i iz tablice iznosi: 16

C je druga znamenka i iz tablice iznosi: 3072

2 je treća znamenka i iz tablice iznosi: 8192

+

što ukupno daje: 11294

Znači da je broj $2C1E_{16}$ po iznosu jednak 11294_{10}.

Pretvorba dekadskih u heksadecimalne brojeve

Ponovo ćemo se služiti tablicom. Na primjer, treba pronaći heksadecimalni broj koji odgovara broju 20500.

Najbliži broj iz tablice je u stupcu znamenke broj 3, u retku 5 i iznosi 20480.

Imamo dakle:

5 20480

Još nam manjka 20 do broja 20500, pa tražimo u stupcu znamenke 2. Tamo je svaki broj veći od 20 pa idemo u stupac znamenke 1. Pošto smo stupac 2 preskočili imamo:

50 20480

U stupcu 1 najbliži broju 20 je broj 16 iz retka 1 pa imamo:

501 20480 + 16 = 20496

Još nam manjka 4 što pronalazimo u stupcu 0 i retku 4. Imamo:

5014 20496 + 4 = 20500

I konačno:

$5014_{16} = 20500_{10}$

Pokažimo pretvorbu na još jednom primjeru. Potražimo heksadecimalni ekvivalent dekadskog broja 55555.

U stupcu treće znamenke imamo 53248 u retku D. Dakle imamo:

D 53248, te nam još manjka

55555 — 53248 = 2307.

U stupcu druge znamenke u retku 9 nalazimo 2304, pa imamo:

D9 53248 + 2304 = 55552 pa nam još manjka:

55555 — 55552 = 3.

U stupcu prve znamenke nemamo tako malog broja pa idemo u stupac nulte znamenke, u treći redak. Za sada imamo D90, dakle:

55552 + 3 = 55555

i konačno:

$D903_{16} = 55555_{10}$

Pretvorba uz pomoć tablice zahtijeva malo vježbe nakon koje neće biti nikakvih problema pri pretvorbi.

NAREDBE SYS I USR

Postoji nekoliko metoda upotrebe BASIC-a i strojnog jezika na C64, uključujući i specijalne naredbe kao dijelove CBM BASIC-a, kao i lokacije tipaka na stroju. Pet je glavnih načina upotrebe strojnog jezika iz BASIC-a na C64. Oni su:

1. BASIC SYS naredba
2. BASIC USR funkcija
3. Promjena jednog od RAM I/O vektora
4. Hardware interrupt
5. Promjena CHRGET rutine

1. BASIC naredba SYS uzrokuje skok (JUMP) na strojni potprogram lociran na adresi X. Rutina mora završavati s instrukcijom RTS (ReTurn from Subroutine — povratak iz subrutine). To će vratiti kontrolu natrag u BASIC. Parametri se općenito vode između strojnih rutina i BASIC-programa uz BASIC PEEK i POKE naredbe, i njihove strojne ekvivalente. SYS-komanda je najkorisnija metoda kombiniranja BASIC-a i strojnog jezika. Može se upotrijebiti mnogo naredbi SYS u programu, svaka za različitu (ili istu) strojnu rutinu.

2. BASIC funkcija USR(X) prenosi kontrolu na strojni potprogram pohranjenu na adresi na lokacijama 785 i 786 (adresa je pohranjena u standardnom formatu »niži byte — viši byte«). X-vrijednost prolazi prema strojnom potprogramu preko floating point akumulatora 1, lociranog počevši od adrese $61. Vrijednost može biti vraćena natrag u BASIC-program smještanjem u akumulator pokretnog zareza. Strojna rutina mora završiti s instrukcijom RTS za povratak u BASIC.

Ova naredba razlikuje se od SYS, jer je potrebno postaviti indirektni vektor. Različit je i format kroz koji varijabla provodi. Indirektni vektor mora se izmijeniti ako se primjenjuje više od jedne strojne rutine.

3. Bilo koja input/output (ulazno-izlazna) rutina ili interna rutina u BASIC-u kojima se može pristupiti preko tablice vektora locirane na memorijskoj stranici 3 može se nadomjestiti ili ispraviti korisničkim kodom. Svaki od dvobytenih vektora sadrži niži i viši byte adrese koju upotrebljava operacijski sustav. KERNAL VECTOR rutina je najpouzdaniji način promjene bilo kojeg vektora, ali pojedini vektor može se

promijeniti pomoću POKE. Novi vektor će pokazivati na rutinu koju je pripremio korisnik koja je zamišljena kao nadomjestak ili proširenje standardne sistemske rutine. Kada se odgovarajuća BASIC-komanda izvrši, izvršit će se korisnikova rutina. Ako je nakon izvršenja korisničke rutine nužno izvesti i normalnu sistemsku rutinu, korisnikov program mora JMP (JUMP — skočiti) na adresu prije sadržanu u vektoru. Ako ne, rutina mora završavati sa RTS da prenese kontrolu natrag u BASIC.

4. HARDWARE INTERRUPT (IRQ VEKTOR) može se izmijeniti. Svake 1/50 sekunde operativni sistem prenosi kontrolu na rutinu specificiranu tim vektorom. KERNAL to normalno služi za vremenovanje, pretraživanje tastature, itd. Ako se upotrijebi ta tehnika, može se uvijek prenijeti kontrola na normalnu IRQ-rutinu, osim ako je rutina kojom je nadomještena ta rutina, nije pripremljena za rukovanje čipom CIA.

Ova metoda je korisna za zadatke koji se moraju izvoditi paralelno s BASIC-programom, ali je zapreka u tome što je veoma teška.

5. CHRGET se upotrebljava s BASIC-om za dobivanje svakog znaka. To čini jednostavnim dodavanjem nove BASIC-naredbe. Svaka nova naredba mora biti izvršena strojnim potprogramima koje je pripremio korisnik. Opća je značajka ove metode u tome da se označi što će se desiti prije bilo koje nove komande. Nova rutina CHRGET će tražiti specijalni karakter (znak). Ako nijedan nije nađen, kontrola se prenosi na normalnu BASIC CHRGET-rutinu. Ako je specijalni karakter tu, nova naredba će se interpretirati i izvršiti strojnim programom. To skraćuje posebno vrijeme za izvršenje, izazvano potrebom traženja dodatnih naredbi. Tehnika se često naziva **wedge.**

NOVE NAREDBE U STROJNOM KODU

Program EXTENDED(prošireni) BASIC

Program EXT. BASIC formira niz rutina koje stvaraju set novih BASIC-naredbi na C64. Te nove naredbe pokrivaju područje zvuka i grafike koje je prilično zapostavljeno u Commodoreovom originalnom BASIC-interpreteru. Program se sastoji od 2 dijela: loadera (punjača) i strojnog koda. Prvi dio upišite i spremite na vrpcu. Zatim upišite drugi dio i startajte ga s RUN. Program će ispitati da li su podaci korektno upisani i ako postoje greške, javit će ih. Ako je sve u redu, dobit ćete poruku da pripremite kazetu. Vrpcu namjestite tako da se ovi podaci usnime odmah iza programa 1. Za svaki slučaj, nakon toga spremite na drugu kazetu i sam program 2 za rezervu. EXT. BASIC ćete startati tako da vrpcu premotate na početak programa 1 i pritisnete tipke SHIFT i RUN/STOP. Kompjuter će sam učitati oba dijela programa.

Naredbe kojima raspolažete:

VOLume — namješta jakost zvuka
sintaksa: SYS49664,"VOL"a
a — (Ø—15)

ENVelope — kontrola ADSR-parametara (vidi poglavlje Zvuk na C64)
sintaksa: SYS49664,"ENV"k,a,d,s,r
k — kanal (1—3)
a,d,s,r — (Ø—15)

SILEnce — reset čipa SID, isključenje zvuka
SYS49664,"SIL"

PULse — širina pulsnog vala
SYS49664,"PUL"k,š
k — kanal (1—3)
š — širina (Ø—4Ø95)

WAVe — odabir valne forme
SYS49664,"WAV"k,f
k — kanal (1—3)
f — forma/Ø-trokut
1 — pila
2 — kvadrat
3 — šum

VOIce — uključuje ili isključuje pojedini kanal i namješta notu
SYS49664,"VOI"k,b
k — kanal (1—3)
b — nota (Ø—96) — Ø — isklj.

SOUnd — uključuje kanal s određenom notom za određeno vrijeme
SYS49664,"SOU"k,b,v
k — kanal (1—3)
b — nota (Ø—96)
v — vrijeme (Ø—32767) — 1 sek = 1ØØ

POInter — postavlja pokazivač spritea
SYS49664,"POI"s,l
s — sprite (Ø—7)
l — lokacija (Ø—16383)

ENAble — uključuje ili isključuje sprite
SYS49664,"ENA"s,u
s — sprite (Ø—7)
u — uklj.-isklj. (1—Ø)

SPRite — pozicionira sprite na koordinate x-y
SYS49664,"SPR"s,x,y
s — sprite (Ø—7)
x — (Ø—511)
y — (Ø—255)

Sprite COLor — određuje boju spritea
SYS49664,"SCO"s,b,m
s — sprite (Ø—7)
b — boja (Ø—15)
m — uklj.-isklj. (1—Ø) multikolor moda (ovaj parametar nije nužan)

SIZe — povećavanje spritea
SYS49664,"SIZ"s,px,py
s — sprite (Ø—7)
px — povećanje u x-osi (Ø — ne, 1 — da)
py — povećanje u y-osi (Ø — ne, 1 — da)

PRIoritet — govori da li sprite ide ispod ili iznad teksta
SYS49664,"PRI"s,p
s — sprite (Ø—7)
p — Ø — ispod, 1 — iznad

SpriteMUlticolor — određuje boje spritea u mc modu
SYS49664,"SMU"m1,m2
m1 — boja (Ø—15)
m2 — boja (Ø—15)

Lijevi SCroll — pomak cijelog ili dijela ekrana ulijevo
SYS49664,"LSC"gl,dl
gl — gornja linija (Ø—24)
dl — donja linija (Ø—24)
uvjet: dl veći od gl

Desni SCroll — pomak cijelog ili dijela ekrana udesno
SYS49664,"RSC"gl,dl
gl — gornja linija (Ø—24)
dl — donja linija (Ø—24)

BORder — određuje boju bordera — ruba
SYS49664,"BOR"b
b — boja (Ø—15)

BACkground — boja pozadine
SYS49664,"BAC"b
b — boja (Ø—15)

BLAnk — uključuje i isklj. ekran (radi malog povećanja brzine pri računanju)
SYS49664,"BLA"e
e — (Ø — isklj., 1 — uklj. ekran)

SCReen — širina i dužina ekrana
SYS49664,"SCR"k,r
k — kolona (38/4Ø)
r — redova (24/25)

POSition — postavlja x i y scroll registre — služi za scroll u svim smjerovima po 1 karakter
SYS49664,"POS"x,y
x — (Ø—7) normalno Ø
y — (Ø—7) normalno 3

CURset — položaj kursora
SYS49664,"CUR"x,y
x — (Ø—39)
y — (Ø—24)

JOYstick — čita vrijednosti palice za igru
USR(Ø),"JOY"p,s
p — port (Ø — port 2, 1 — port 1)
s — prekidač (Ø — gore, 1 — dolje, 2 — lijevo, 3 — desno, 4 — pucanje)
— funkcija vraća Ø ako je prekidač otvoren ili 1 ako je zatvoren

COLision — detektira sprite/sprite ili sprite/data sudar
USR(Ø),"COL"ss,sd
Ø — sprite
1 — data

```
5 REM PROGRAM 1
10 IFFL=0THENFL=1 LOAD"EXTENSION 64",1,1
20 PRINT "[illegible]** CBM 64 BASIC V2 + NOVE NAREDBE**"
30 FB=FRE(0)-65536*(FRE(0)<0)+205
40 PRINTTAB(7)FB"BASIC BYTA SLOBODNO"
50 POKE785,0:POKE786,194
60 NEW

READY.
```

```
10 REM**EXTENDED BASIC
80 FS$="C846":REM SLOBODNA MEMORIJA POCINJE OVDJE
90 GOTO5000
100 REM IMENA & ADRESE
110 DATA VOL,C200,ENV,C295,SIL,C2E4,PUL,C2F2,WAV,C32D,VOI,C460,SOU,C4BB
120 DATA POI,C510,ENA,C538,SPR,C55B,SCO,C590,SIZ,C5CA,PRI,C604,SMU,C629
130 DATA LSC,C685,RSC,C6CD,BOR,C715,BAC,C723,BLA,C731,SCR,C751,POS,C799
140 DATA CUR,C7C6,JOY,C7F2,COL,C81D
290 DATA*
999 REM M.KOR
```

```
1000 DATA 20,73,00,A2,00,20,73,00,9D,A7,02,E8,E0,03,D0,F5,1694
1010 DATA A2,00,BD,00,C6,CD,A7,02,D0,27,BD,01,C6,CD,A8,02,1921
1020 DATA D0,1F,BD,02,C0,CD,A9,02,D0,17,20,73,00,C9,22,D0,1819
1030 DATA F9,8A,4A,AA,BD,00,C1,85,A7,BD,01,C1,85,A8,6C,A7,2272
1040 DATA 00,E8,E8,E8,E8,BD,00,C0,D0,04,E8,4C,12,C2,A2,0B,6C,2090
1050 DATA 00,03,98,48,8A,48,20,86,AE,20,AA,B1,84,A7,85,A8,1756
1060 DATA 68,AA,68,A8,A5,A7,60,A6,A9,C5,A9,B0,03,A2,0E,6C,2100
1070 DATA 00,03,C8,FF,F0,07,C8,84,A9,C5,A9,B0,F0,60,00,00,2076
1080 DATA 20,52,C2,A2,00,A0,0F,20,67,C2,AD,18,D4,29,F0,05,1669
1090 DATA A7,8D,18,D4,60,20,52,C2,A2,01,A0,03,20,67,C2,AA,1773
1100 DATA CA,A9,D4,85,FE,A9,05,85,FD,E0,00,F0,0B,18,A5,FD,2447
1110 DATA 69,07,85,FD,CA,4C,A9,C2,A0,00,84,02,20,52,C2,A2,1803
1120 DATA 00,A0,0F,20,67,C2,0A,0A,0A,0A,8D,A7,02,20,52,C2,1162
1130 DATA A2,00,A0,0F,20,67,C2,0D,A7,02,A4,02,91,FD,C8,C0,1804
1140 DATA 02,D0,D7,60,A2,19,A9,00,9D,FF,D3,CA,D0,FA,20,73,2307
1150 DATA 00,60,20,52,C2,A2,01,A0,03,20,67,C2,AA,CA,A9,D4,1812
1160 DATA 85,FE,A9,02,85,FD,E0,00,F0,0B,18,A5,FD,69,07,85,2106
1170 DATA FD,CA,4C,06,C3,20,52,C2,A9,0F,C5,A8,B0,03,4C,6D,1953
1180 DATA C2,A0,00,A5,A7,91,FD,A5,A8,C8,91,FD,60,20,52,C2,2419
1190 DATA A2,01,A0,03,20,67,C2,85,FD,AA,A9,00,9D,A9,02,20,1740
1200 DATA 52,C2,A2,00,A0,03,20,67,C2,AA,A9,10,E0,00,F0,05,1754
1210 DATA 0A,CA,4C,4C,C3,A6,FD,9D,A9,02,60,79,00,C9,2C,F0,2000
1220 DATA 01,60,20,52,C2,A2,00,A0,01,20,67,C2,86,FE,A9,01,1614
1230 DATA A6,FD,E0,01,F0,05,0A,CA,4C,72,C3,85,FD,A5,FE,F0,2581
1240 DATA 09,AD,17,D4,05,FD,8D,17,D4,60,A9,FF,38,E5,FD,85,2242
1250 DATA FE,2D,17,D4,0D,17,D4,60,00,00,00,00,00,00,00,00,1006
1260 DATA 0C,01,1C,01,2D,01,3E,01,51,01,66,01,7B,01,91,01,606
1270 DATA A9,01,C3,01,DD,01,FA,01,18,02,38,02,5A,02,7D,02,1142
1280 DATA A3,02,CC,02,F6,02,23,03,53,03,86,03,BB,03,F4,03,1317
1290 DATA 30,04,70,04,B4,04,FB,04,47,05,98,05,ED,05,47,06,1109
1300 DATA A7,06,0C,07,77,07,EB,07,61,00,E1,08,68,09,F7,09,1265
1310 DATA 8F,0A,30,0B,DA,0B,8F,0C,4E,0D,18,06,EF,0E,D2,0F,1203
1320 DATA C3,10,C3,11,D1,12,EF,13,1F,15,60,16,B5,17,1E,19,1337
1330 DATA 9C,1A,31,1C,DF,1D,A5,1F,87,21,06,23,A2,25,DF,27,1505
1340 DATA 3E,2A,C1,2C,6B,2F,3C,32,39,35,63,38,BE,3B,4B,3F,1257
1350 DATA 0F,43,8C,47,45,4B,BF,4F,7D,54,83,59,D6,5E,79,64,1537
1360 DATA 73,6A,C7,70,7C,77,97,7E,1E,86,18,8E,8B,B6,7E,9F,1956
1370 DATA FA,A0,06,B3,AC,BD,F3,C8,E6,D4,0F,E1,F8,EE,2E,FD,3002
1380 DATA 20,52,C2,A2,01,A0,03,20,67,C2,AA,CA,86,FC,AB,D4,2102
1390 DATA 85,FE,A9,00,85,FD,E0,00,F0,0B,18,A5,FD,69,07,85,2104
1400 DATA FD,CA,4C,76,C4,20,52,C2,A2,00,A0,60,20,67,C2,C9,2181
1410 DATA 00,D0,0A,A0,04,A6,FC,BD,AA,02,B1,FD,60,A0,00,AA,1905
1420 DATA CA,8A,0A,AA,BD,A0,C3,91,FD,8D,A1,C3,C8,B1,FD,A6,2771
1430 DATA FC,BD,AA,02,A0,04,09,81,91,FD,60,20,60,C4,20,52,1718
1440 DATA C2,A5,A7,D0,07,A5,A8,D0,03,4C,93,C4,38,A5,A7,EB,2325
1450 DATA B1,85,A7,A5,A8,E9,00,85,A8,A2,F9,A0,00,C8,D0,FD,2400
1460 DATA E0,D0,FA,20,E1,FF,F0,03,4C,C1,C4,60,48,A9,81,E0,2472
1470 DATA 00,F0,05,0A,CA,4C,EF,C4,AA,A0,00,60,F0,06,0A,11,1003
1480 DATA FD,91,FD,60,86,FB,30,AB,FF,E5,FB,31,FD,91,FD,60,2080
1490 DATA 20,52,C2,A2,00,A0,07,20,67,02,85,FB,20,52,C2,A5,1023
1500 DATA A8,A2,00,A0,3F,20,67,C2,06,A7,26,A8,06,A7,26,A0,1640
1510 DATA A5,A0,A6,FB,BD,F0,07,60,20,52,C2,A2,00,A0,07,20,1927
1520 DATA 67,C2,85,FB,20,52,C2,A2,00,A0,01,20,67,C2,A6,FB,2058
1530 DATA A0,15,84,FD,A0,D0,84,FE,4C,EC,C4,20,52,C2,A2,00,2290
1540 DATA A0,B7,20,67,C2,85,FC,20,52,C2,A5,A0,A2,00,A0,01,1845
1550 DATA 20,67,C2,A6,FC,A0,18,84,FD,A0,D0,64,FE,20,EC,C4,2526
1560 DATA 06,FC,A5,A7,A6,FC,BD,00,D0,20,52,C2,BD,01,D0,60,2143
1570 DATA 20,52,C2,A2,00,A0,07,20,67,C2,85,FB,20,52,C2,A2,1820
1580 DATA 00,A0,0F,20,67,C2,A6,FB,BD,27,D0,20,70,00,C9,2C,1723
1590 DATA FB,B1,60,20,52,C2,A2,00,A0,01,20,67,C2,A6,FB,A0,1074
1600 DATA D0,84,FE,A0,1C,84,FD,4C,EC,C4,20,52,C2,A2,00,A0,2305
1610 DATA 07,20,67,C2,85,02,20,82,C2,A2,00,A0,01,20,67,C2,1431
1620 DATA A6,02,A0,D0,84,FE,A0,1D,84,FD,20,EC,C4,20,52,C2,2260
1630 DATA A2,00,A0,01,20,67,C2,A0,D0,84,FE,A0,17,84,FD,A6,2146
1640 DATA 02,4C,EC,C4,20,52,C2,A2,00,A0,07,20,67,C2,85,FB,1860
1650 DATA 20,52,C2,A2,00,A0,B1,20,67,C2,A6,FB,A0,D0,84,FE,2131
1660 DATA A0,13,84,FD,49,B1,4C,EC,C4,A0,00,84,02,20,82,C2,1756
1670 DATA A2,00,A0,0F,20,67,C2,A6,02,9D,25,D0,E8,06,02,E0,1828
1680 DATA 02,D0,EA,60,20,52,C2,A2,00,A0,10,20,67,C2,8D,A7,1031
1690 DATA 02,20,52,C2,A2,00,A0,18,20,67,C2,8D,A8,02,CD,A7,1660
1700 DATA 02,B0,03,4C,6D,C2,60,A9,00,85,FB,A9,04,85,FC,C8,1959
1710 DATA 00,D0,01,60,18,A5,FB,69,28,85,FB,A5,FC,69,00,85,1923
1720 DATA FC,00,4C,6F,C6,20,44,C6,AC,A7,02,20,67,C6,AE,A7,2086
1730 DATA 02,A0,01,B1,FB,00,91,FB,C8,18,A5,FC,69,D4,85,FC,2466
1740 DATA B1,FB,00,91,FB,38,A5,FC,E9,D4,05,FC,C8,C8,C8,20,2895
1750 DATA D0,E1,00,A9,20,91,FB,18,A5,FB,69,28,05,FB,A5,FC,2552
1760 DATA 69,00,05,FC,E0,EC,A0,02,90,C7,F0,C5,60,20,44,C6,2302
1770 DATA AC,A7,02,20,67,C6,AE,A7,02,A0,26,B1,FB,C8,91,FB,2233
1780 DATA 88,18,A5,FC,69,D4,85,FC,B1,FB,C8,91,FB,38,A5,FC,2776
1790 DATA E9,D4,85,FC,88,88,C0,FF,D0,E1,C0,A9,20,01,FB,18,2003
1800 DATA A5,FB,69,28,85,FB,A5,FC,69,00,85,FC,E0,EC,A0,02,2490
1810 DATA 90,C7,F0,C5,60,20,52,C2,A2,00,A0,0F,20,67,C2,0D,1991
1820 DATA 20,D0,60,20,52,C2,A2,00,A0,0F,20,67,C2,8D,21,D0,1692
1830 DATA 60,20,52,C2,A2,00,A0,01,20,67,C2,C9,00,F0,0B,AD,1679
1840 DATA 11,D0,29,EF,8D,11,D0,60,AD,11,D0,09,10,8D,11,D0,1756
1850 DATA 60,20,52,C2,A2,26,A0,20,20,67,C2,C9,27,D0,03,4C,1660
1860 DATA 6D,C2,C9,26,D0,0B,AD,16,D0,29,F7,8D,16,D0,4C,79,2020
1870 DATA C7,AD,16,D0,09,00,8D,16,D0,20,52,C2,A2,10,A0,1B,1669
1880 DATA 20,67,C2,C9,10,D0,09,AD,11,D0,29,F7,8D,11,D0,60,1810
1890 DATA AD,11,D0,09,08,8D,11,D0,60,20,52,C2,A2,00,A0,07,1514
1900 DATA 20,67,C2,05,FB,AD,16,D0,29,F0,05,FB,8D,16,D0,20,2004
1910 DATA 52,C2,A2,00,A0,07,20,67,C2,05,F9,AD,11,D0,29,F0,2005
1920 DATA 05,FB,8D,11,D0,60,20,52,C2,A2,00,A0,27,20,67,C2,1716
1930 DATA 85,02,20,82,C2,A2,00,A0,10,20,67,C2,AA,A4,02,4C,1539
1940 DATA F0,FF,A0,00,B1,FD,E0,00,F0,05,4A,CA,4C,E6,C7,29,2376
1950 DATA 01,60,20,52,C2,A2,00,A0,01,20,67,C2,48,20,52,C2,1437
1960 DATA A2,00,A0,04,20,67,C2,AA,68,A0,A9,DC,85,FE,A9,00,2042
1970 DATA 85,FD,20,E4,C7,49,01,A0,A9,00,6C,05,00,20,52,C2,1677
1980 DATA A2,00,A0,07,20,67,C2,48,20,52,C2,A2,00,A0,01,20,1393
1990 DATA 67,C2,A0,60,AA,A9,D0,85,FE,AB,1E,85,FD,20,E4,07,2547
2000 DATA A8,A9,00,6C,05,00,AF,A4,AC,0C,FF,84,FE,0A,EF,20,2151
2010 DATA*
2999 REM SPREMANJE IMENA I ADRESA
3000 A1=49152 A2=49408
3010 READA$:IFA$="*"THEN3050
3020 FORN=1TO3:POKEA1-1+N,ASC(MID$(A$,N,1)):NEXT:POKEA1+3,0
3030 READH$:GOSUB20000:POKEA2+1,INT(D/256):POKEA2,D-PEEK(A2+1)*256
3040 A1=A1+4:A2=A2+2:GOTO3010
3050 POKEA1-1,255:RETURN
4000 A=49664
4010 CS=0:READH$:IFH$="*"THENRETURN
4020 GOSUB20000:CS=CS+D:POKEA,D:A=A+1
4030 FORN=1TO15:READH$:GOSUB20000
4040 POKEA,D:A=A+1:CS=CS+D:NEXT
4050 READC:IFC<>CSTHENPRINT"GRESKA U PODACIMA U "PEEK(63)+256*PEEK(64):END
4060 GOTO 4010
5000 PRINT"[illegible]CBM64EXTENSION"
5010 PRINT"[illegible]PRICEKAJTE,PUNJENJE PODATAKA"
5020 GOSUB3000
5030 GOSUB4000
5040 PRINT"PRIPREMITE KAZETU...I PRITISNITE TIPKU":POKE198,0:WAIT198,1
5045 A$="64 EXT.BASIC"
5050 FORN=1TO12:POKE670+N,ASC(MID$(A$,N,1)):NEXT
5060 POKE780,12:POKE781,167:POKE782,2
5070 SYS65469
5080 POKE253,0:POKE254,192:POKE780,253
5090 H$=FS$:GOSUB20000
5100 POKE782,INT(D/256):POKE781,D-PEEK(782)*256
5110 SYS65496
9999 END
20000 REM HEX-DEC KONVERZIJA
20010 D=0
20020 IFH$>""THENFORI=1TOLEN(H$):W=ASC(MID$(H$,I,1))-48:D=D*16+W+(W>9)*7
20030 IFH$>""THENNEXT
20040 RETURN
READY.
```

KORISNI PROGRAMI

SPRITE CONTROL

Program S. CONTROL pomaže pri vezivanju rutine za kontrolu joysticka i programa za definiciju spriteova. U program je uključeno i nekoliko naredbi koje omogućuju promjene na ekranu. Naredbe u programu su:

I — postavlja sprite ispred podataka
B — postavlja sprite iza podataka
H — postavlja sprite u mod HR (visoke rezolucije)
M — postavlja sprite u mod MC (multikolor)
X — povećava sprite u osi x
Y — povećava sprite u osi y
U — smanjuje sprite u osi y
V — smanjuje sprite u osi x
O — isključuje sprite
S — uključuje sprite
P — mijenja boju pozadine
Q — mijenja boju bordera
C — mijenja boju spritea
R — skok na »hladni« start računala
W — čišćenje ekrana
E — završava program
A — mijenja sprite MC Ø
D — mijenja sprite MC 1

Da biste dobili puni prioritet spriteova, program izlistajte prije starta. Podaci o spriteovima

treba da su smješteni negdje pri kraju inicijalnog programa, a liniju 7Ø promijenite u 7Ø FOR N=Ø TO 62:READ Q:POKE 12288+N, Q:NEXT. Pri upotrebi palice za igru primijenite PORT 1.

```
0 CLR
1 A=5:B=5:C=1 REM SPRITE CONTPOLEP
2 X=100:Y=100
5 POKE 650,PEEK(650)OR128
10 V=53248
11 POKE V+39,7
20 POKE V+21,1
30 POKE V+32,B
40 POKE V+33,C
50 POKE V+46,5
60 POKE 2040,192
70 FOR N=0 TO 62:POKE 12288+N,255:NEXT
95 POKE V,X:POKE V+1,Y
100 IF Y<50 THEN Y=Y+5
105 IF Y>230 THEN Y=Y-5
110 IF X<25 THEN X=X+5
115 IF X>250 THEN X=X-5
130 O=PEEK(56321)
140 IF O=254 THEN Y=Y-5
150 IF O=246 THEN Y=Y-5:X=X+5
160 IF O=247 THEN X=X+5
170 IF O=245 THEN X=X+5:Y=Y+5
180 IF O=253 THEN Y=Y+5
190 IF O=249 THEN X=X-5:Y=Y+5
200 IF O=250 THEN X=X-5:Y=Y-5
300 PEM*DOPISNICKE FUNKCIJE*
310 GET A$
320 IF A$="I" THEN POKE V+27,0
325 IF A$="B" THEN POKE V+27,1
330 IF A$="M" THEN POKE V+28,1
335 IF A$="A" THEN POKE V+28,1:POKE V+37,C+1
340 IF A$="D" THEN POKE V+28,1:POKE V+38,C+1
345 IF A$="H" THEN POKE V+28,0
350 IF A$="E" THEN END
355 IF A$="X" THEN POKE 53277,PEEK(53277)OR(210)
360 IF A$="Y" THEN POKE 53271,PEEK(53271)OR(210)
365 IF A$="U" THEN POKE 53271,PEEK(53271)AND(255-210)
370 IF A$="V" THEN POKE 53277,PEEK(53277)AND(255-210)
375 IF A$="O" THEN POKE 53269,PEEK(53269)AND(255-210)
380 IF A$="S" THEN POKE 53269,PEEK(53269)OR(210)
385 IF A$="P" THEN C=C+1:POKE 53281,C+1
390 IF A$="G" THEN B=B+1:POKE 53280,B+1
395 IF A$="C" THEN A=A+1:POKE V+39,A
400 IF A$="W" THEN PRINT"J"
405 IF A$="P" THEN SYS64738
1000 GOTO 95

READY.
```

COMMODORE 64 TOOLKIT

Program TOOLKIT pruža nekoliko korisnih mogućnosti na C64. Kao prvo daje mogućnost ispisivanja kompletnih naredbi pritiskom na jednu ili dvije tipke, tako da se dobiva efekt sličan SPECTRUMOVOM upisivanju naredbi. Funkcijske tipke su definirane u tablici, a ako želite možete ih i mijenjati u okviru linija 6Ø i 7Ø. Ako na kraju riječi (koja ne smije biti dulja od 8 karaktera) stavite znak (←), to daje automatski efekt pritiska na tipku RETURN.

Uz ispisivanje naredbi, TOOLKIT pruža i mogućnost štampanja sadržaja ekrana na pisač, omogućuje zaustavljanje listanja pritiskom na tipku (←), i nastavak s RETURN, te također pruža mogućnost kontrole kursora preko JOYSTICK PORT 2. Primjer je dan u malom programu. Sam program TOOLKIT upišite u računalo i spremite na vrpcu, jer bi u slučaju krivo upisanih podataka kompjuter mogao »zablokirati«. Nakon startanja programa s naredbom RUN i učitavanja strojnog kôda, aktivirajte program naredbom SYS 156 × 256. Program ostaje u memoriji do isključenja.

TABLICA FUNKCIJSKIH TIPKI

F1	LIST	F5	READ
F2	RUN&RETURN	F6	DATA
F3	SAVE	F7	GOSUB
F4	VERIFY&RET.	F8	RETURN

COMMODORE & F1 INPUT
COMMODORE & F3 RESTORE
COMMODORE & F5 PEEK(
COMMODORE & F7 POKE
SHIFT & CTRL & F1 CHR$(
SHIFT & CTRL & F3 MID$(
SHIFT & CTRL & F5 LEFT$(
SHIFT & CTRL & F7 RIGHT$(
COMM. & CTRL & F1 ASC(
COMM. & CTRL & F3 STR$(
COMM. & CTRL & F5 VAL(
COMM. & CTRL & F7 TAB(
CTRL & F1 promjena boje BORDER-a
CTRL & F3 promjena boje pozadine
CTRL & F5 kopija ekrana na pisač (veliki karakteri i mali razmak linija)
CTRL & F7 kopija ekrana na pisač (mali karakteri i normalan razmak linija)

```
READY.

5 SYS156*256 PRINT"♫"
10 SC=1024:CL=55296 X=0 Y=0
20 GET A$:IF A$="" THEN 20
30 IF A$="■"AND X>0 THEN X=X-1
40 IF A$="■"AND X<39 THEN X=X+1
50 IF A$="T"AND Y>0 THEN Y=Y-1
60 IF A$="■"AND Y<24 THEN Y=Y+1
70 POKE SC+X+(Y*40),42 POKE CL+X+(Y*40),1
80 GOTO 20

READY.
```

```
READY.

10 FOR I=0 TO 12 READ K$ K=LEN(K$)
15 FOR J=0 TO K-1 W=ASC(MID$(K$,J+1,1))
20 IF W=95 THEN W=13
25 POKE 157*256+I*8+J,W
30 NEXT J
35 FOR K=K TO 7 POKE 157*256+I*8+K,0 NEXT K
40 NEXT I
50 FOR I=0 TO 252 READ A POKE 156*256+I,A NEXT
55 FOR I=160 TO 496 READ A POKE 157*256+I,A NEXT
60 DATALIST,RUN←,INPUT,CHR$(,SAVE,VERIFY←,RESTORE,MID$(
65 DATAREAD,DATA,PEEK(,LEFT$(,GOSUB,RETURN,POKE,RIGHT$(
70 DATAASC(,STR$(,VAL(,TAB(
100 DATA72,138,72,152,72,169,156,133,52,133,56,169,0,133,51,133,55
110 DATA120,169,85,141,20,3,169,156,141,21,3,58
120 DATA162,0,189,50,156,240,8,32,210,255,232,224,255,208,243
130 DATA76,231,156,170,104,96
140 DATA147,32,32,32,32,32,32,32,42,42,42,42,32,67,66,77,32
150 DATA54,52,32,84,79,79,76,75,73,84,32,42,42,42,42,13,13,0
155 DATA72,138,72,152,72,165,197,197,2,208,5,133,2,76,220,157,133,2
160 DATA201,57,208,3,76,220,157,201,7,144,3,76,220,157,201,3
170 DATA176,3,76,220,157,174,141,2,224,4,208,3,76,207,156,201,3
180 DATA240,6,160,136,136,136,136,152,10,10,10,10,10,224,1,208,3,24,105,0
185 DATA224,2,208,3,24,105,16,224,5,208
187 DATA3,24,105,24,224,6,208,5,74,74,24,105,120
190 DATA133,252,169,157,133,253,160,0,177,252,153,119,2
200 DATA208,192,0,208,246,132,198,104,168,104,170,104,76,49,234
210 DATA201,4,208,3,76,160,157,201,5,208,3,76,176,157
220 DATA201,6,200,3,76,211,158,76,227,150
225 DATA120,169,192,141,40,3,169,157,141,41,3
226 DATA169,4,133,255,88,104,168,104,170,104,96
230 DATA173,32,208,41,15,170,232,138,41,15,141,32,208,76,220,157
240 DATA173,33,208,41,15,170,232,138,41,15,141,33,208,76,220,157
250 DATA165,197,201,57,240,3,76,237,246,165,197,201,1,240,6
260 DATA201,63,240,243,208,244,169,0,133,190,76,237,246
270 DATA173,0,220,201,127,208,3,76,199,156,201,126,208,5,162,145
280 DATA76,22,158,201,125,208,5,162,17,76,22,158,201,123,208,5,162,157
290 DATA76,22,158,201,119,208,5,162,29,76,22,158,201,111,208,5,162,19
300 DATA76,22,158,76,199,156,130,166,255,202,134,255,240,3,76,199,156,162,4
310 DATA134,255,160,1,132,198,141,119,2,76,199,156
320 DATA72,152,72,130,72,120,169,0
330 DATA32,109,255,173,24,208,160,255
340 DATA162,4,41,2,240,2,160,7
350 DATA169,4,32,186,255,32,192,255
360 DATA162,4,32,201,255,176,113,169
370 DATA0,133,251,169,4,133,252,32
380 DATA237,255,134,254,134,253,132,255
390 DATA160,0,169,14,32,210,255,177
400 DATA251,201,128,144,9,41,127,72
410 DATA169,18,32,210,255,104,201,34
420 DATA208,2,169,39,201,32,144,0
430 DATA201,64,144,7,9,128,208,3
440 DATA24,105,64,32,210,255,169,146
450 DATA32,210,255,200,208,2,230,252
460 DATA198,254,208,203,198,255,240,27
470 DATA169,0,32,210,255,169,13,32
480 DATA210,255,169,15,32,210,255,169
490 DATA14,32,210,255,166,253,134,254
500 DATA24,144,172,169,13,32,210,255
510 DATA32,231,255,104,170,104,160,104
520 DATA88,76,220,157
530 DATA169,14,141,106,150,141,103,158
540 DATA169,0,141,168,158,76,47,158
550 DATA169,15,141,106,158,141,103,150
560 DATA141,168,150,76,47,158

READY.
```

COMMODORE 64 DELETE

Program DELETE omogućava brisanje samo određenih linija u programu, preciznije linija između dva linijska broja, koje zadate Commodoreu. U brisanje su uključene i te dvije granične linije. Uz to, program pruža i mogućnost autonumeracije linija pri pisanju programa. Pri tome birate početni broj linije i korak za slijedeće linije, koje se zatim normalno upisuju i završavaju tipkom RETURN.

COMMODORE 64 BLOCK SAVE

Budući da C64 ne posjeduje naredbu SAVE CODE, koja bi spremala dio BASIC-memorije, ovaj program pruža tu mogućnost. Može se spremiti bilo koja količina memorije, a treba upisati početnu i završnu adresu bloka koji se sprema, njegovo ime, broj aparata na koji se spremaju informacije (1 — kazetofon, 8 — disk-jedinica), te sekundarnu adresu (1 ili Ø). Ako upišete 1, informacije će se učitavati s normalnom naredbom LOAD, a ako upišete Ø učitava se naredbom LOAD" ",1,1. Kad upišete program B. SAVE startajte ga s RUN i upišite tražene podatke. Da bi se napokon podaci, program ili strojni kôd spremili na vrpcu ili disk, unesite direktnu naredbu SYS 49152 i pritisnite RETURN.

```
READY.

95 END
62995 STOP
63000 POKE 53280,1:POKE 53281,1
63002 POKE 36879,27:PRINT"":PRINT"AUTO NUMERACIJA N":PRINT"BRISANJE D"
63005 PRINT"KRAJ F"
63010 GET A$:IF A$<>"N"AND A$<>"D"AND A$<>"F"THEN 63010
63020 IF A$="N"THEN 63100
63030 IF A$="F"THEN END
63040 INPUT"OD";A:INPUT"DO ";B:PRINT""
63050 IF A<0 OR B<0 OR A>62999 OR B>62999 THEN 63020
63060 IF A>B THEN 63000
63070 PRINT""A
63080 PRINT"A="A+1":B="B":GOTO 63060
63090 POKE 198,3:POKE 631,19:POKE 632,13:POKE 633,13:END
63100 INPUT"START";A:INPUT"KORAK";B:IFA<0ORB<1ORA>62999ORB>62999THEN63020
63110 IF A>62999 THEN 63000
63120 PRINT""A;
63130 SYS(65487):PRINT
63160 PRINT"A="A+B":B="B":GOTO63110"
63170 POKE 198,3:POKE 631,19:POKE 632,13:POKE 633,13:END

READY.
```

```
READY.

1 INPUT"IME BLOKA";F$
2 F$=LEFT$(F$,8)
3 FORN=1 TO LEN(F$):POKE819+N,ASC(MID$(F$,N,1)):NEXT
4 INPUT"STARTNA ADRESA";SA
5 INPUT"KRAJNJA ADRESA";EA
6 INPUT"KAZETOFON(1)ILI DISK(8)";DN
7 INPUT"MARKER (1) ILI (0)";ES
8 II=49152
9 I=49152
10 READ A$
20 IF A$="*" THEN 200
30 AL$=LEFT$(A$,1):AR$=RIGHT$(A$,1)
40 IF ASC(AL$)<65 THEN D=VAL(AL$)*16:GOTO 60
50 D=(ASC(AL$)-55)*16
60 IF ASC(AR$)<65 THEN D=D+VAL(AR$):GOTO 80
70 D=D+ASC(AR$)-55
80 POKE I,D:I=I+1:GOTO 10
100 DATA A9,00,A2,00,A0,FF,20,BA,FF,A9,01,A2,34,A0,03,20,BD,FF,A9,FF,85,FB
110 DATA A9,FF,85,FC,A2,00,A0,FF,A9,FB,20,D8,FF,60,*
200 POKE II+3,DN:POKE II+5,ES
210 POKE II+18,LEN(F$)
220 A=INT(SA/256):POKE II+19,SA-A*256:POKE II+23,A
230 A=INT(EA/256):POKE II+27,EA-A*256:POKE II+29,A

READY.
```

COMMODORE 64 SLOWLIST

Poznato je da C64 izlistava program prevelikom brzinom da bi se mogao čitati, čak i kada je pritisnuta tipka CTRL, koja malo usporava listanje. Zbog toga ova kratka rutina omogućava kontrolu listanja pomoću funkcijskih tipki. Ako je pritisnuta F1, listanje se zaustavlja i linije se mogu pregledavati. Ako se pritisne F3, listanje je toliko usporeno da se vidi ispisivanje pojedinačnog karaktera u liniji. Usput će se tako pojaviti i linije u BASIC-u koje su bile skrivene s karakterima za brisanje. Brzina sporog listanja ovisi o varijabli u liniji 3Ø, a najbolje vrijednosti su između 230 i 255. Program se nakon upisivanja i startanja može izbrisati naredbom NEW, no efekt usporenog listanja ostaje sve do isključenja računala.

```
10 REM TIPKA F1 ZA PAUZU U LISTANJU,TIPKA F3 ZA USPORENO LISTANJE
20
30 POKE 251,248 REM BRZINA LISTANJA(230-255 NAJPOGODNIJE VRIJEDNOSTI)
40 FOR S=0 TO 22 READ MC POKE 49152+S,MC NEXT
50 POKE 774,0 POKE 775.192
60 DATA 72,165,197,201.4.240,250,201,5
70 DATA 208,0,165,251,133,162,165,162,48,252,104,76,26,167

READY.
```

COMMODORE 64 AUTO NUMBER

Program AUTO NUMBER omogućava automatsko numeriranje programskih linija pri pisanju programa na C64. Time se štedi i vrijeme i program se urednije ispisuje. Program trebate upisati i startati sa RUN da bi se učitao strojni kôd. Sam program se starta naredbom SYS 49152, zaustavlja se sa SYS 2048 ili pritiskom na tipke RUN STOP/RESTORE, a ponovno stavlja u pogon sa SYS 49166. Tada se numeracija nastavlja od zadnje ispisane i unesene linije. Kad pišete svoj program, ako želite novi broj linije, pritisnite razmaknicu dok je kursor na prvoj poziciji u redu. Kad napišete kompletnu programsku liniju i pritisnete RETURN, automatski će se ispisati slijedeći broj linije i kursor će biti pozicioniran za pisanje.

```
READY.

PRESS PLAY ON TAPE

READY.

10 DATA 169,48,133,2,133,4,133,5
20 DATA 133,6,169,49,133,3,128,169
30 DATA 27,141,28,3,169,192,141,21
40 DATA 3,80,96,165,211,201
50 DATA 1,208,6,169,60,197
60 DATA 197,240,3,76,49,234
70 DATA 169,0,133,211,169,40,162,4
80 DATA 213,2,208,5,202,224,255,208
90 DATA 247,232,134,198,160,0,136,181
100 DATA 2,153,119,2,208,202,224,255
110 DATA 208,245,230,198,169,32,153,119
120 DATA 2,162,1,246,2,169,58,213
130 DATA 2,208,9,169,48,149,2,232
140 DATA 224,5,208,239,76,49,234
150 K=0
160 FOR X=49152 TO 49256
170 READ H:K=K+H:POKEX,H
180 NEXT X
190 IF K<>13846 THEN PRINT"GRESKA U UNOSU PODATAKA-PROVJERITI!":END
200 REM**SYS (49152) ZA STARTCLOSE**
210 REM ** SYS (2048) ILI RUN-STOP/RESTORE ZA PREKID AUTO NUMERACIJE **
220 REM ** SYS (49166) ZA NASTAVAK **
230 SYS(49152)
240 NEW

READY.


READY.
```

COMMODORE 64 MERGE

Program MERGE dodaje BASIC-u C64 komandu koja mu omogućava spajanje dva dijela programa koji se zasebno učitavaju, ali nema brisanja prvog programa iz memorije. Svakako treba pripaziti da se brojevi linija ne preklapaju, jer bi tada drugi program izbrisao linije prvog programa koje imaju iste brojeve. Program MERGE upišite i spremite na kazetu, a zatim ga startajte sa RUN, da bi se kôd upisao iznad RAM koji se inače upotrebljava za BASIC. Nakon toga programe možete učitavati naredbom SYS 40705 »ime programa« i povezivati ih. To je vrlo korisno pri pisanju programa, ako imate biblioteku potprograma i rutina koje se tako mogu jednostavno učitati i povezati, a ne prepisivati ručno tastaturom.

```
0 PRINT"": POKE 53281,0:POKE 53280,0:REM**MERGE**
1 POKE 55,0 POKE 56,159 CLR A=40705 FORH=A TO A+78:READ MC:POKE H,MC:NEXT:NEW
2 DATA 169,0,133,10,32,212,225,165,43
3 DATA 72,165,44,72,56,165,45,233,2,133
4 DATA 43,165,46,233,0,133,44,169,0,133
5 DATA 185,166,43,164,44,169,0,32,213
6 DATA 255,176,14,134,45,132,46,32,51
7 DATA 165,104,133,44,104,133,43,96,170
8 DATA 201,4,144,244,240,10,104,133,44
9 DATA 104,133,43,24,108,0,3,164,186
10 DATA 136,240,209,200,239
11 REM UCITAVANJE SA SYS 40705"IME PR."

READY.
```

COMMODORE 64 KEYPOLY

KEYPOLY pretvara CBM-64 u mali polifoni instrument na kojemu možete čak i svirati ako znate. Za sviranje služe tipke u drugom redu odozgo, tj. Q,W,E,R,T itd., a pritiskom na funkcijske tipke možete birati jedan od 3 valna oblika ili kraj programa. Program upišite i prije startanja ga spremite na vrpcu. To je potrebno za slučaj ako ste načinili neku grešku pri upisu. Program se starta naredbama RUN/RETURN i potrebno mu je kraće vrijeme da spremi kôd u memoriju. Nakon toga možete okušati svoje muzičko umijeće.

```
10 PRINT"    ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌"
20 PRINT"    ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌"
30 PRINT"    ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌"
40 PRINT"    ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌ ▌"
50 PRINT"    ▌ | | | | | | | | | | | | |"
60 PRINT"    Q W E R T Y U I O P @ * ↑"
70 PRINT"        F1 VALNI OBLIK TROKUT"
80 PRINT"        F3 VALNI OBLIK PILA"
90 PRINT"        F5 VALNI OBLIK CETVRT."
100 PRINT"        F7 KRAJ"
110 PRINT"         PUNJENJE STR.KODA"
120 IFPEEK(49152)=32ANDPEEK(49153)=93THEN180
130 FORC=49152TO49655STEP8 CK=C-256*INT(C/256) FORCC=0TO7 READM CK=(CK+M)AND255
140 POKE C+CC,M NEXT
150 IF(PEEK(63)+256*PEEK(64))<>CTHENPRINT"  LINIJE"C" NEMA" POKE49152,96 END
160 READ M IF M<>CK THEN 210
170 NEXT
180 AD=000 SR=195 POKE49270,AD POKE49289,SR
190 PL=000 PH=000 POKE49256,PL:POKE49267,PH
200 PRINT"        " SYS49152:END
210 PRINT"  GRESKA U LINIJI"PEEK(63)+256*PEEK(64) POKE49152,96 END
49152 DATA32,93,192,32,20,192,32,169,2
49160 DATA192,32,1,193,162,50,160,0,30
49160 DATA136,200,253,202,200,250,32,199,224
49176 DATA193,76,3,192,120,162,7,169,170
```

```
49184 DATA127,141,0,220,72,173,1,220,218
49192 DATA157,244,193,104,56,106,202,16,94
49200 DATA248,80,162,63,169,255,157,252,154
49208 DATA193,202,16,250,162,0,160,0,15
49216 DATA189,244,193,134,251,162,7,10,230
49224 DATA72,144,5,169,0,153,252,193,36
49232 DATA200,104,202,16,242,166,251,232,213
49240 DATA192,64,208,228,96,162,24,169,207
49248 DATA0,157,0,212,202,16,250,169,70
49256 DATA160,141,2,212,141,9,212,141,98
49264 DATA16,212,169,15,141,3,212,141,253
49272 DATA10,212,141,17,212,169,0,141,254
49280 DATA5,212,141,12,212,141,19,212,58
49288 DATA169,249,141,6,212,141,13,212,255
49296 DATA141,20,212,169,15,141,24,212,54
49304 DATA169,255,141,52,3,141,53,3,201
49312 DATA141,54,3,169,64,141,55,3,22
49320 DATA96,174,52,3,48,24,189,252,230
49328 DATA193,240,8,169,0,157,252,193,108
49336 DATA76,198,192,173,55,3,141,4,2
49344 DATA212,169,255,141,52,3,174,53,227
49352 DATA3,48,24,189,252,193,240,8,133
49360 DATA169,0,157,252,193,76,227,192,194
49368 DATA173,55,3,141,11,212,169,255,211
49376 DATA141,53,3,174,54,3,48,24,212
49384 DATA189,252,193,240,8,169,0,157,160
49392 DATA252,193,76,0,193,173,55,3,161
49400 DATA141,18,212,169,255,141,54,3,217
49408 DATA96,162,0,134,251,173,52,3,103
49416 DATA48,11,173,53,3,48,6,173,11
49424 DATA54,3,48,1,96,189,130,193,210
49432 DATA168,185,252,193,240,3,32,39,112
49440 DATA193,232,224,23,208,223,96,173,124
49448 DATA52,3,16,3,76,80,193,173,124
49456 DATA53,3,16,3,76,105,193,152,137
49464 DATA141,54,3,189,153,193,141,14,176
49472 DATA212,189,176,193,141,15,212,173,95
49480 DATA55,3,9,1,141,18,212,96,95
49488 DATA152,141,52,3,189,153,193,141,00
49496 DATA0,212,189,176,193,141,1,212,188
49504 DATA173,55,3,9,1,141,4,212,102
49512 DATA96,152,141,53,3,189,153,193,60
49520 DATA141,7,212,189,176,193,141,0,155
49528 DATA212,173,55,3,9,1,141,11,213
49536 DATA212,96,57,60,14,15,9,22,101
49544 DATA23,17,20,38,31,25,38,39,103
49552 DATA33,36,46,41,44,54,55,49,246
49560 DATA52,135,134,162,223,62,193,107,196
49568 DATA60,57,99,190,75,15,12,69,225
49576 DATA191,125,131,214,121,115,199,124,100
49584 DATA33,35,37,39,42,44,47,50,247
49592 DATA53,56,59,63,67,71,75,79,195
49600 DATA84,89,94,100,106,112,119,173,45
49608 DATA255,193,240,6,169,16,141,55,251
49616 DATA3,96,173,254,193,240,6,169,62
49624 DATA32,141,55,3,96,173,253,193,139
49632 DATA240,6,169,64,141,55,3,96,230
49640 DATA173,0,194,240,6,104,104,169,190
49648 DATA0,133,198,96,0,0,0,0,155

READY.
```

COMMODORE 64 INT-BLUES

Program INT-BLUES pruža mogućnost da C64 svira određenu melodiju dok korisnik potpuno nezavisno od toga upisuje nešto (program, podatke itd.) u njega. Uz to je vrlo pogodan za testiranje i vježbu rada sa čipom SID. Sam program ima mogućnost spremanja do 255 nota, koje se pohranjuju na lokacijama od 49200 do 50955. Strojni kôd je spremljen na adresama 51000—51106. Tako na raspolaganju za programiranje bilo čega drugoga, ostaje slobodnih najmanje 36 K. Ako sami želite mijenjati neke parametre da biste čuli rezultat, držite se opaski koje su napisane u REM-linijama i koje vam daju osnovne informacije na što se koji POKE odnosi. Ako želite isključiti melodiju, utipkajte POKE 75Ø,1 ili resetirajte kompjuter. Prva metoda je bolja, jer omogućuje kasnije ponovno aktiviranje melodije s POKE 75Ø,Ø. Za mijenjanje valnih oblika možete u adrese 753, 754 i 755 upisivati vrijednosti 17, 33 ili 65. Za dužinu trajanja nota mijenjajte varijablu u adresi 752. Npr., vrijednost 30 će dati pauzu od 1/2 sekunde između nota itd. Možete se poigrati i s vrijednostima u linijama 130—150 i 300—350. Pazite da broj nota ostane isti, jer biste u protivnom morali mijenjati i brojeve u linijama 100—120 koje učitavaju te podatke.

```
10 DATA 173,236,2,240,3,76,49,234,206,239,2,208,248,173,248,2
20 DATA 141,239,2,174,251,0,189,49,192,201,255,240,70,141,1,212
30 DATA 189,82,193,141,0,212,189,176,194,141,8,212,189,188,195,141
40 DATA 7,212,189,224,196,141,15,212,189,12,198,141,14,212,232,234
50 DATA 142,251,0,162,0,141,4,212,141,11,212,141,18,212,173,241
60 DATA 2,141,4,212,173,242,2,141,11,212,173,243,2,141,18,212
70 DATA 76,49,234,169,0,141,251,0,76,69,199
80 FOR I=51000 TO 51106 READA POKE I,A NEXT
90 DIM HT(10),LT(10),HM(8),LM(8),HB(8),LB(8)
100 FOR A=0 TO 10 READ HT(A),LT(A) NEXT
110 FOR A=0 TO 8 READ HM(A),LM(A) NEXT
120 FOR A=0 TO 8 READ HB(A),LB(A) NEXT
130 DATA0,0,32,94,38,126,43,52,48,127,51,97,57,172,64,188,68,149,76,252,255,255
140 DATA 0,0,12,216,19,63,21,154,24,63,25,177,28,214,32,94,38,126
150 DATA 0,0,6,108,8,23,9,159,12,32,12,216,14,107,16,47,19,63
200 FOR A=0 TO 103 READ B POKE 49200+A,HT(B) POKE 49500+A,LT(B) NEXT
210 FOR A=0 TO 103 READ B POKE 49800+A,HM(B) POKE 50100+A,LM(B) NEXT
220 FOR A=0 TO 103 READ B POKE 50400+A,HB(B) POKE 50700+A,LB(B) NEXT
299 REM**NOTE KANAL1**
300 DATA 1,0,1,0,3,0,4,0,7,7,6,6,7,6,4,4,1,0,1,0,3,0,4,0,7,7,6,6,7,6,9,9
301 DATA 1,0,1,0,3,0,4,0,7,7,6,6,7,6,4,2,3,3,3,3,2,2,3,3,7,0,9,0,0,0,9,9
302 DATA 6,6,6,6,0,0,0,0,5,5,3,3,1,1,1,1,3,3,4,4,2,2,2,2,2,2,2,2,2,2,0,0
303 DATA 0,0,0,0,0,0,10,10
309 REM**NOTE KANAL2**
310 DATA 0,0,0,0,0,0,0,0,5,5,6,6,7,6,2,2,0,0,0,0,0,0,0,0,5,5,6,6,7,6,0,0
311 DATA 0,0,0,0,0,0,0,0,5,5,6,6,7,6,2,2,0,0,0,0,0,0,0,0,5,5,7,6,6,7,7,7
312 DATA 6,6,6,6,0,0,0,0,0,0,0,0,0,0,0,0,3,3,4,4,2,2,2,2,1,1,1,1,1,1,0,0
313 DATA 0,0,0,0,0,0,0,0
319 REM**NOTE KANAL3**
320 DATA 2,2,2,2,0,0,0,0,7,7,0,0,0,0,4,4,2,2,2,2,0,0,0,0,7,7,0,0,0,0,0,0
321 DATA 2,2,2,2,0,0,0,0,7,7,0,0,4,4,3,3,0,0,0,0,3,3,0,0,0,0,0,0,0,0,0,0
322 DATA 6,6,6,6,0,0,0,0,5,5,0,0,2,2,2,2,0,0,0,0,3,3,3,3,1,1,1,1,0,0,0,0
323 DATA 0,0,0,0,0,0,0,0
500 S=54272
509 REM**VOLUME/CUT OFF/FILTER**
510 POKE S+24,31 POKE S+22,128 POKE S+23,244
519 REM**ADSP KANAL1**
520 POKE S+5,0 POKE S+6,240
529 REM**ADSP KANAL2**
530 POKE S+12,85 POKE S+13,133
539 REM**ADSP KANAL3**
540 POKE S+19,10 POKE S+20,137
549 REM**VALNI OBLIK KANAL1**
550 POKE 753,17
559 REM**VALNI OBLIK KANAL2**
560 POKE 754,65 POKE S+10,0
569 REM**VALNI OBLIK KANAL3**
570 POKE 755,33
579 REM**TRAJANJE NOTE(1/60 SEK)**
580 POKE 752,0
589 REM**ZVUK UKLJ.(0) ISKLJ.(1)**
590 POKE 750,0
599 REM**USMJ.VEKTOR NA 51000**
600 POKE 788,56 POKE 789,199

READY.
```

PREGLED TIPSKIH PROGRAMA

HELP 64 plus

Program HELP 64 sastoji se od četiri dijela:

NAREDBE ZA OBRADU BASIC-PROGRAMA

Ovaj dio sadržava 22 naredbe koje pojednostavljuju pisanje i promjenu BASIC-programa. Ima naredbi za spajanje više programa, promjenu broja linija u programu, brisanje više linija i traženje riječi u programu. Osim toga neke naredbe olakšavaju traženje grešaka u programima, tj. praćenje toka programa, izvođenje korak po korak i ispis svih varijabli.

STROJNI MONITOR — DISASSEMBLER

Za rad sa strojnim programima ima 13 naredbi koje omogućuju pregled i promjenu memorije strojnih programa. Strojni monitor osim toga može raditi i s programima u memoriji diska.

DOS — NAREDBE ZA RAD S DISKOM

Program sadržava i 8 naredbi za rad s disketom. Ove naredbe omogućuju da pojedine operacije s diskom izvodimo pritiskom na jednu tipku.

ASSEMBLER

Integralni dio programa je i dvoprolazni assembler za mikroprocesore 6500 do 6515. Assembler se može upotrebljavati samo s diskom.

SUPERGRAPHIK 64

Program nakon učitavanja zauzme 8 K memorije, te proširuje BASIC s pedesetak novih instrukcija za manipulaciju slikama i zvukom.

GRAPH 64

Izvrstan matematički program koji iscrtava grafove zadanih funkcija u zadanom koordinatnom sustavu.

STAT 64

Proširenje BASIC-a za poslovne primjene (statistički dijagrami itd.).

LOGOS 64

Jedan od boljih tekst-procesora koji imaju mogućnost rada i s kazetofonom (ne samo s diskom).

EASYSCRIPT

Zacijelo najbolji tekst-procesor za Commodore 64, a radi samo s diskom.

GRAPH BASIC 64

Dosad najbolje proširenje BASIC-a C64, barem što se tiče naredbi za grafiku i muziku.

SPRITEMAKER

Služi za definiranje oblika spriteova.

SYNTHY 64

Proširenje BASIC-a muzičkim naredbama.

SAM/RECITER

Program za sintezu govora. Commodore priča engleskim naglaskom, a na žalost i izgovorom.

MUSIC CONSTRUCTION SET

Muzički program. Joystickom se postavljaju note u muzičkom crtovlju, a zatim se i sviraju.

SUPER 64 MON

Ima ugrađen assembler i disassembler, kao i mnogo drugih pogodnosti koje omogućuju pisanje i editiranje strojnih programa.

MASTER ADRESS

Program koji služi prvenstveno za poslovne primjene, a služi kao baza podataka s adresama.

IGRE

SUMMER GAMES

Igra je izvrsna simulacija 8 sportskih disciplina. Izvanredna grafika i animacija likova.

SOCCER II

Od ove simulacije nogometa ima i desetine boljih programa, ali SOCCER II je zacijelo najbolji od programa u kojima mogu igrati dvojica (jedan protiv drugoga).

BC'S QUEST FOR TIRES

Igra koja je često recenzirana i svakoj recenziji uspoređena s crtanim filmom. Postoji i QUEST FOR TIRES 2 (na disku) koji je nekoliko puta bolji . . .

DAVID'S MIDNIGHT MAGIC

Igra koja je prvobitno rađena za Apple, a zatim prerađena i za C64. Vjerojatno je najbolja simulacija flipera.

GHOSTBUSTERS

Istjerivatelji duhova sišli su s filma i s gramofonske ploče u kompjutersku igru, a grafika i zvuk nimalo ne zaostaju za njima.

STRIP POKER

S Commodoreom se može i kartati i za to ima dosta programa. U ovoj igri kartate s jednom od djevojaka, i to u svlačenje.

BEACH HEAD

Bombe, topovi, avioni, ratni brodovi — to je ono što se nalazi u ovoj nadasve dobro izrađenoj igri.

LAZY JONES

Glavni junak ove igre, Jones, nalazi se u hotelu prepunom videoigara. Dakako u hotelu ima mnoštvo soba, a u svakoj sobi nova igra. Vrlo zabavno.

BOULDERDASH

Šesnaest spilja s dijamantima. U svakoj od njih treba ih pokupiti određen broj, a za to je potrebno razmišljanje, vještina i brzina. Tko u igru »zagrize«, taj će ostati nekoliko noći uz ekran.

SUICIDE EXPRESS

Brza, arkadna igra s izuzetnom grafikom. O rezultatu ne morate brinuti jer je u igru uvedena i sinteza govora.

SAUCER ATTACK

Leteći tanjuri napadaju Bijelu kuću u Washingtonu, a vi je morate braniti. Izuzetna grafika!

DECATHLON

Deset atletskih disciplina. Po kvaliteti i originalnosti igra je vrlo primamljiva, ali pravi je neprijatelj joysticka.

BRUCE LEE

Sve je kao i u filmovima.

ERIK THE VIKING

Jedna od najboljih igara avanture. Mnoštvo dobrih screenova.

FORBIDDEN FOREST

Zbog naziva, zastrašujuće glazbe, paukova, kostura, duhova i ostalog što se u njoj susreće, igra je dobila epitet igre strave.

COLOSSUS CHESS

Dosad najbolji šahovski program za Commodore 64.

FALCON PATROL II

U ovoj fantastičnoj igri morate se oduprijeti napadima protivničke avijacije.

ŠTO DALJE?

UPUTE KORISNICIMA ZA DALJI RAD S C64

Kraj knjižice posvećen je onima koje ne zanima samo igranje na računalu već bi svoj rad željeli unaprijediti. Mnogi korisnici će poželjeti usavršiti svoje stručno i praktično znanje, tim više što svako računalo zahtijeva dublje proučavanje, pa tako i C64.

Ozbiljniji korisnici neće se pri programiranju zadovoljiti samo BASIC-om ili strojnim kôdom već će nastojati raditi s drugim programskim jezicima (FORTH-om, FORTRAN-om, LOGO-om, PASCAL-om itd.) koji se upotrebljavaju pomoću prevodioca-kompajlera (compileri). Na tržištu se nalaze u obliku programske podrške snimljene na kazetofonskim vrpcama ili disketama.

Dakle, više programske jezike potrebno je interpretirati ili kompajlirati računalu C64 kako bi ono razumjelo naše zahtjeve. BASIC se interpretira, a kompajliranjem se mogu primijeniti ostali programski jezici pomoću odgovarajućih programa.

BASIC radi na tzv. konverzacijski način, a dubljom analizom može se ustanoviti da je njegova osnovica u programskim jezicima ALGOL-u i FORTRAN-u, naravno uz dodatke koji omogućavaju konverzacijski način rada. Programi koji omogućavaju konverzacijski rad na računalu nazivaju se interpreteri.

Konverzacijski način naziva se ponekad i interaktivni način. Što to znači? To je kada zadate naredbu računalu, a on je istog momenta prevede i izvede. Rad je tada nalik na razgovor čovjeka s računalom. Odavde i dolazi naziv »konverzacijski«.

Ostali jezici su kompajleri (compileri), odnosno neki viši programski jezici moraju se najprije prevesti na strojni jezik, a zatim se on pomoću asemblera prevodi u binarni kôd. Tu se uočava nedostatak viših programskih jezika jer kompajliranje ne daje posebno djelotvoran program u strojnom jeziku, tj. tako dobiveni program zauzima mnogo memorije, a duže je i vrijeme izvođenja. Ni za Commodore ne mogu se pronaći tako dobri kompajleri da loše napisan program u nekom višem programskom jeziku prevede u dobar program u strojnom jeziku. Također je potrebno da kompajleri budu univerzalni, tj. da mogu prevesti bilo koji program na izvornom jeziku i potom ga prevesti u strojni jezik. Zbog potrebne univerzalnosti, memorijski prostor ne iskorištava se optimalno kao ni vrijeme odvijanja programa. To je u biti slično kada ne znamo neki strani jezik pa moramo plaćati prevodioca. Time imamo korist, ali i štetu, upravo kao u prevodilačkim programima na računalu.

Uz programe-kompajlere obično treba nabaviti i neke programe za dijagnosticiranje grešaka. Traženje, uočavanje i ispravljanje grešaka (debugging) vrlo je često najznačajniji dio rada koji je ukupno potreban pri pravljenju programa.

Dakle, kako računalo ne može neposredno shvatiti BASIC, ono ga mora pretvoriti u sebi razumljive nizove logičkih nula i jedinica (binarni kôd). To znači da su viši programski jezici sporiji nego npr. strojni kôd ili asembler. U stvarnosti to rezultira time da se npr. u nekoj igri pomični objekti kreću sporije, a u matematičkim se proračunima sporije dolazi do rezultata.

Za početnike ove razlike u brzini nisu važne. Od svih viših programskih jezika BASIC se najlakše može naučiti i služiti se njime. Zato ga nalazimo kao osnovni jezik u gotovo svim kućnim i džepnim računalima, ali isto tako i u nekim velikim kompjuterskim sustavima. BASIC pruža vrlo dobar uvod u ozbiljnije programiranje, pa je i zato dobro došao. Velika prednost BASIC-a je i to što korisnik može neposredno za tastaturom napraviti svoj program. Za druge jezike, pa čak i za PASCAL, koji je isto tako vrlo pogodan za učenje, potrebno je prvo na papiru zamisliti i postaviti program pomoću dijagrama tokova. Dijagrami tokova su zapravo logičke sheme kako će se koja operacija izvesti i njezin rezultat usmjeriti dalje. Upotreba »specijaliziranih« programskih jezika znatno povećava mogućnosti računala C64.

Što se dopunske literature tiče, nju treba tražiti među inozemnim naslovima bilo da je riječ o proučavanju samog računala, hardwarea ili softwarea općenito, ili složenih projekata primjenom računala.

Austrija, Italija i SR Njemačka su zemlje u koje se dosta često putuje, pa zato sa svakog putovanja uz traperice, rižu i kavu donesite nešto za vaše računalo.

Oni koji često ne putuju mogu literaturu i manje hardverske dijelove naručivati poštom. Literaturu je najbolje i najsigurnije naručivati s pouzdanih adresa, a kao prvi kontakt s inozemnim izdavačima neka bude njihov katalog.

Odabrali smo nekoliko adresa izdavača:

Bernard Babani Publishing Ltd.
The Grampians, Shepherds Bush Road,
London W6 7NF, England

— ima na raspolaganju veliki izbor jeftinih i praktičnih knjiga iz područja računala i mikroelektronike po cijeni manjoj od 3 funte.

DATA BECKER, Merowingerstr. 30,
4000 Düsseldorf 1, BRD

— izdaje literaturu prosječne cijene od oko 50 DM, ali ne isporučuje pošiljke jeftinije od 200 DM.

McGraw — Hill Book Company GmbH
Landemanbogen 136, 2000 Hamburg 63, BRD,

Franzis — Verlag, 8000 München 37
Postfach 31 01 20, BRD

i

Vogel-Buchverlag KG, Postfach 6740
8700 Würzburg 1, BRD

— odlikuju se stručnim naslovima i najkvalitetniji su izdavači na polju primjene računala u svakodnevnom životu.

Za hardverski orijentirane vlasnike dajemo pregled priključnica računala C64, te im savjetujemo da sheme raznih dodataka i naprava traže od spomenutih izdavača — u njihovim naslovima kao i u mnogim časopisima kao npr.: Funkschau, Computer Schau, MC, Run, Elektor, ELO itd.

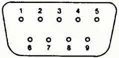
Nabavom dobre stručne literature, što raznovrsnijih programa i elektroničkih dodataka može se znatno povećati praktična primjena računala C64. Nastojte što više programa ili hardverskih dodataka izvesti sami. Trud će vam se višestruko isplatiti.

Također nastojte što više ljudi uz sebe zainteresirati za kreativnu primjenu računala. Neka vam ne bude teško odgovarati na postavljena pitanja pa i na ona »najgluplja«. Svakom pitanju je uzrok težnja za znanjem. Isto tako i vi se nemojte sramiti pitati nešto što ne znate. Nije sramota ne znati, sramota je ne htjeti znati.

Radeći vašim Commodoreom 64 uvijek imajte na umu da je pri radu s njime

— jedino ograničenje vaša mašta.

Kontrolni port



I

Pin	Signal	
1	JOYA0	
2	JOYA1	
3	JOYA2	
4	JOYA3	
5	POT AY**	
6	BUTTON A/LP*	
7	+5V	MAX. 100mA
8	GND	
9	POT AX**	

II

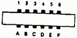
Pin	Signal	
1	JOYB0	
2	JOYB1	
3	JOYB2	
4	JOYB3	
5	POT BY**	
6	BUTTON B	
7	+5V	MAX. 100mA
8	GND	
9	POT BX**	

Priključnica za module



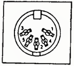
Pin	Signal	Pin	Signal
Z	GND	22	GND
Y	CA0	21	CD0
X	CA1	2D	CD1
W	CA2	19	CD2
V	CA3	18	CD3
U	CA4	17	CD4
T	CA5	16	CD5
S	CA6	15	CD6
R	CA7	14	CD7
P	CA8	13	DMA
N	CA9	12	BA
M	CA1D	11	ROML
L	CA11	1D	I/O 2
K	CA12	9	EXROM
J	CA13	8	GAME
H	CA14	7	I/O 1
F	CA15	6	Dot Clock
E	Φ 2	5	CR/W
D	NMI	4	IRQ
C	RESET	3	+5V
B	ROMH	2	+5V
A	GND	1	GND

Priključnica za kazetofon



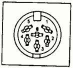
Pin	Signal
A-1	GND
B-2	+5V
C-3	CASSETTE MOTOR
D-4	CASSETTE READ
E-5	CASSETTE WRITE
F-6	CASSETTE SENSE

Audio / Video



Pin	Signal
1	LUMINANCE
2	GND
3	AUDIO OUT
4	VIDEO OUT
5	AUDIO IN

Serijski ulaz/izlaz



Pin	Signal
1	SERIAL SRQIN
2	GND
3	SERIAL ATN IN/OUT
4	SERIAL CLK IN/OUT
5	SERIAL DATA IN/OUT
6	RESET

Korisnički (User) port

Pin	Signal	
1	GND	
2	+5V	MAX. 100 mA
3	RESET	
4	CNT1	
5	SP1	
6	CNT2	
7	SP2	
8	PC2	
9	SER. ATN IN	
10	9 VAC	MAX. 100 mA
11	9 VAC	MAX. 100 mA
12	GND	
A	GND	
B	FLAG2	
C	PB0	
D	PB1	
E	PB2	
F	PB3	
H	PB4	
J	PB5	
K	PB6	
L	PB7	
M	PA2	
N	GND	



DODATAK

YU-ZNAKOVI ZA C64

Program YU-ZNAKOVI omogućuje vlasnicima Commodorea 64 da pri radu s ekranom kao izlaznom jedinicom, što je najčešće u upotrebi, dobiju specijalne YU-znakove Ž, Č, Đ, Ć, Š kojih nema na tastaturi.

Kako?

Nakon učitavanja i startanja programa potrebno je pričekati 1,5 min dok se na ekranu ne pojavi READY.

Potom pritisak na tipku ∂ izaziva pojavljivanje slova Ž na ekranu, lijeva uglata zagrada označava Č, V označava Đ, desna uglata zagrada označava Ć, a & označava Š.

Kako je to postignuto?

Da bismo predefinirali neki karakter najprije treba kopirati karakter ROM-a u RAM što je postignuto programskim linijama od 9 do 143.

Preostali dio programa predefinira karaktere što je regulirano FOR-petljom, potprogramom i DATA-naredbama.

```
3 REM ****************************
5 REM *** YU ZNAKOVI ZA C-64 ***
6 REM ****************************
7 !
8 REM *KOPIRANJE CHARACTER ROM-A U RAM*
9 !
10 POKE 56333,127
20 POKE 56,67
30 POKE 1,51
40 Z=24576
50 X=53248
60 Y=PEEK(X):X=X+1
70 IF X=57344 THEN 100
80 POKE Z,Y:Z=Z+1
90 GOTO 60
```

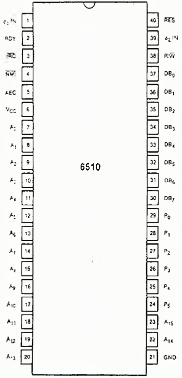
```
100 POKE 1,53
110 POKE 56576,6
120 POKE 53272,25
130 POKE 648,68
140 POKE 56333,129
143 :
145 REM *PREDEFINIRANJE CHARACTERA*
147 :
150 B=24576
160 A1=B
170 A2=B+27*8
180 A3=B+28*8
190 A4=B+29*8
200 A5=B+30*8
210 RESTORE
220 FOR I=A1 TO A1+7
230 GOSUB 500
240 NEXT I
250 FOR I=A2 TO A2+7
260 GOSUB 500
270 NEXT I
280 FOR I=A3 TO A3+7
290 GOSUB 500
300 NEXT I
310 FOR I=A4 TO A4+7
320 GOSUB 500
330 NEXT I
340 FOR I=A5 TO A5+7
350 GOSUB 500
360 NEXT I
370 END
375 :
380 DATA 48,15,124,8,16,32,124,0
390 DATA 48,16,56,68,64,68,56,0
400 DATA 60,34,34,114,34,34,60,0
410 DATA 8,16,56,68,64,68,56,0
420 DATA 48,16,56,64,56,4,120,0
450 :
500 READ A
510 POKE I,A
520 RETURN
READY
```

MIKROPROCESOR 6510

6510 je jeftini mikrokompjuterski sistem sposoban za rješavanje široke lepeze problema i kontrolu perifernih uređaja. Na čipu se nalazi 8-bitna dvosmjerna ulazno-izlazna sabirnica. 16--bitna adresna sabirnica omogućava direktno adresiranje 65535 memorijskih adresa. Interna arhitektura procesora je identična procesoru MOS Technology 6502, da bi se osigurala softverska kompatibilnost.

Karakteristike 6510:

- 8-bitni dvosmjerni ulazno-izlazni port (sabirnica)
- napajanje +5 volti
- 8-bitno paralelno procesiranje
- 56 instrukcija
- decimalna i binarna aritmetika
- trinaest modova adresiranja
- programibilni STACK pokazivač
- varijabilna dužina STACK-a
- mogućnost interupta (prekida)
- 8-bitna dvosmjerna sabirnica podataka
- direktan pristup memoriji do 64K
- sabirnica kompatibilna sa M6800
- brzina takta 1 Mhz i 2 Mhz
- upotrebljiv s memorijama bilo kojeg tipa ili brzine



MIKROPROCESOR 6510

TEHNIČKE KARAKTERISTIKE COMMODORE 64

MIKROPROCESOR	Commodore MOS 6510
MEMORIJA	64K RAM, 38K za BASIC 20K ROM — operacijski sustav, BASIC V2
PROGRAMSKI JEZICI	COMMODORE BASIC V2, dodatno: Simon's BASIC, PASCAL, FORTRAN, LOGO, COMAL, ASSEMBLER, PILOT
TASTATURA	QWERTY, 63 standardna znaka ASCII, 62 grafička znaka, 4 funkcijske tipke za 8 programabilnih funkcija
GRAFIKA	Visoka razlučivost (320 × 200 točaka), 16 boja
SPRITE GRAFIKA	8 spriteova sastavljenih od 21 × 24 točke
ZNAKOVI	25 redova sa po 40 znakova
EKRAN	TV-prijemnik (c/b ili kolor) ili monitor

GENERATOR ZVUKA	3 kanala, svaki sa 8 oktava, 4 različita oblika valova i krivulja generatora zvuka, različiti filteri, generator šuma
MEĐUSKLOP	RS 232C (V24) za TTL nivo, IEEE 488, mogućnosti raznih proširenja

LITERATURA

1. P. Cornes & A. Cross — Secrets of the Commodore 64, Bernard Babani Publishing Ltd., London 1984.
2. ... — Commodore 64 Handbuch
3. Z. Vistrička — Tajne Commodorea 64, časopis TREND, 1/1985, Zagreb
4. Z. Vistrička — Sprite grafika, časopis TREND, 3/1985, Zagreb
5. Z. Vistrička — Osobna računala, časopis Elektrotehničar, 4/1984, Zagreb
6. Ž. Reljić i B. Makanec — BASIC — kompjuterski jezik, Tehnička knjiga, Zagreb 1984.
7. S. Money — Commodore 64 — Graphics and Sound, Granada, London 1984.
8. ... Commodore 64 Programmer's Reference Guide Commodore Computers, West Chester 1984
9. Časopisi: Commodore microcomputers, Your Computer, Run, razni brojevi i godišta.

SADRŽAJ

PREDGOVOR 3

UVOD 5
O mikroračunalima i mjestu C64 među njima 5
Namjena C64 (područja primjene, dometi, mogućnosti) 7

UPOZNAVANJE I RAD S RAČUNALOM 11
Sustav i upotreba C64 11
— dijelovi i njihova uporaba s kratkim opisom 11
— priključivanje i puštanje u pogon . 13
Tastatura C64 16
Kazetofon 21
Disk-jedinica 22
Pisač (Printer) 24
Moduli 24
Ostala oprema 24

GRAĐA I FUNKCIONIRANJE C64 . . 27
Osnovni pojmovi — bit, byte, ROM, RAM, EPROM 27
Izvedba C64 32
Podjela memorije 32

UVOD U PROGRAMIRANJE 35
BASIC-struktura i namjena programskog jezika 35
Osnovi BASIC-a 37
— kako se pišu programi 37
— brojevi i njihovo pisanje . . . 39
— varijable 40

— programiranje upotrebom instrukcija GOTO, IF ... THEN, FOR ... NEXT 42
— upotreba slučajnih brojeva — instrukcija RND 44
Operacije u BASIC-u 45
— aritmetičke operacije 46
— složene matematičke operacije . . 47
— logičke operacije (AND, OR, NOT) 48
Naredbe i funkcije 49
Nedostaci C64 BASIC-a, mogućnosti upotrebe dodatnog Simon's BASIC-a . . . 59
Instrukcije u Simon's BASIC-u . . . 60
Dojava grešaka 71
GRAFIČKE MOGUĆNOSTI C64 . . . 73
Grafika i boja na C64 73
CHR$ kôdovi i generiranje boje . . . 74
— boja 75
Upotreba PEEK i POKE 76
Sprite-grafika 77
— bojenje spriteova 79
— pokretni spriteovi 80
— proširenje spriteova 81
— sudari spriteova 82
Grafika visoke razlučivosti — grafika ekrana, memorijska i kolor podjela . . . 82
— organizacija ekrana i uporaba . . 83
— primjena boje 85
ZVUK NA C64 87
Zvuk i zvučni efekti 87
Generiranje zvuka 88
— aditivna sinteza 90
Zvučne ovojnice (envelope) 92
Filteri 93
Programiranje zvuka na C64 94
Demonstracijski programi za tvorbu zvučnih efekata 98
STROJNI KÔD 101
Karakteristike 101
Pretvorba heksadecimalnog u dekadski brojevni sustav 102
— pretvorba dekadskih u heksadecimalne brojeve 103
Naredbe SYS i USR 104
Nove naredbe u strojnom kôdu . . . 105
KORISNI PROGRAMI 109
Sprite control 109
C64 Toolkit 110
C64 DELETE 112
C64 Block SAVE 112
C64 Slowlist 113
C64 Auto number 113
C64 Merge 114
C64 Keypoly 114
C64 INT-Blues 115
PREGLED TIPSKIH PROGRAMA . . 117
ŠTO DALJE? 121
Upute korisnicima za dalji rad s C64 . 121
DODATAK 125
YU-znakovi za C64 125
Mikroprocesor 6510 126
Tehničke karakteristike C64 127

Popularna naučno-tehnička biblioteka

KNJIGA DVADESET DRUGA

Zvonimir Vistrička, dipl. inž.
Davor Žunić, inž.

»COMMODORE 64 — UVOD U RAD I PROGRAMIRANJE«

Glavni urednik:
DUBRAVKO MALVIĆ

Tehnički urednik:
VJEKOSLAV BOSNAR

Recenzenti:
RUĐER JENY
Mr STJEPAN SZABO, dipl. inž.

Lektor:
ZORKA HORVATIĆ

Priprema za tisak:
IZDAVAČKI ODJEL NARODNE TEHNIKE HRVATSKE
Zagreb, Dalmatinska 12

Tisak:
»Zadružna štampa« — OOUR Knjigotisak, Zagreb, Dalmatinska 5 i 12